MODELLING AND SIMULATION 2015

THE EUROPEAN SIMULATION

AND

MODELLING CONFERENCE

2015

ESM®'2015

EDITED BY

Marwan Al-Akaidi

and

Aladdin Ayesh

OCTOBER 26-28, 2015

LEICESTER, UNITED KINGDOM

A Publication of EUROISIS-ETI
Cover pictures of Leicester are licensed under the Creative Commons Attribution-Share Alike
The 29th Annual European Simulation and Modelling Conference 2015

LEICESTER, UNITED KINGDOM

OCTOBER 26-28, 2015

Organised by
ETI - The European Technology Institute

Sponsored by
EUROSIS - The European Simulation Society
De Montfort University - Leicester

IEEE – UKRI-SPC

Co-Sponsored by

Ghent University

University of Skovde

Hosted by

The Holiday Inn
Leicester, United Kingdom
EXECUTIVE EDITOR

PHILIPPE GERIL
(BELGIUM)

EDITORS

General Conference Chairs
Dr. Aladdin Ayesh, De Montfort University, Leicester, United Kingdom
Prof. Dr Marwan Al-Akaidi, Vice-Rector Academic Affairs, AOU, Kuwait

Past Conference Chairs
António Carvalho Brito, FEUP - University of Porto, Porto, Portugal
João Manuel Tavares, FEUP - University of Porto, Porto, Portugal

Journal Publication Chairs
Yan Luo, NIST, Gaithersburg, USA
Peter Lawrence, Swinburne University, Australia

ESM Conference Chair
António Carvalho Brito, FEUP - University of Porto, Porto, Portugal

Local Committee Chairs
Duska Rosenberg, De Montfort University, Leicester, United Kingdom
Martin Stacey, De Montfort University, Leicester, United Kingdom
Ali Al-Bayatti, De Montfort University, Leicester, United Kingdom
Francois Siewe, De Montfort University, Leicester, United Kingdom
Neil McBride, De Montfort University, Leicester, United Kingdom
Tom Collins, De Montfort University, Leicester, United Kingdom
Thom Corah, De Montfort University, Leicester, United Kingdom

INTERNATIONAL PROGRAMME COMMITTEE

Methodology and Tools
Jon Andersson, Chalmers Univ. of Technology, Gothenburg, Sweden
Bjorn Johansson, Chalmers Univ. of Technology, Gothenburg, Sweden
Claudia Krull, Otto-von-Guericke University, Magdeburg, Germany
Erik Lindskog, Chalmers Univ. of Techn, Gothenburg, Sweden
Laszlo Muka, Ellassys Consulting Ltd., Budapest, Hungary
J. Manuel Feliz Teixeira, University of Porto, Porto, Portugal
Bert van Beek, Eindhoven University of Technology, Eindhoven, The Netherlands

Discrete Simulation Modeling Techniques and Tools
Track Chair
Renato Natal Jorge, FEUP - University of Porto, Porto, Portugal

IPC
Matthias Becker, University Hannover, Hannover, Germany
Helge Hagenauer, Universitaet Salzburg, Salzburg, Austria
Sophie Hennequin, ENIM, Metz Cedex, France
Bjorn Johansson, Chalmers University of Technology, Gothenburg, Sweden
Goreti Marreiros, Polytechnic of Porto, Porto, Portugal
Stefano Marrone, Seconda Universita di Napoli. Naples, Italy
Guilherme Pereira, Universidade do Minho, Campus de Gaia, Braga, Portugal
Antonella Zanzi, Universita' degli Studi dell'Insubria, Varese, Italia
INTERNATIONAL PROGRAMME COMMITTEE

Simulation and Artificial Intelligence
Helder Coelho, Fac Ciencias, Lisbon, Portugal
Paulo Cortez, University of Minho, Guimareas, Portugal
Martin Hruby, Brno University of Technology, Brno, Czech Republic
Vladimir Janousek, Brno University of Technology, Brno, Czech Republic
Kifaya Qaddoum, Amman, Jordan
Leon Rothkrantz, TU Delft, The Netherlands
Jim Torresen, University of Oslo, Norway
Morched Zeghal, Nat. Res.Council Canada, Ottawa, Canada

Agent Based Simulation
Track Chair
Rosaldo Rossetti, FEUP - University of Porto, Porto, Portuual

IPC
Zisheng Huang, Vrije Universiteit Amsterdam, The Netherlands
Peter Lawrence, Australian Catholic University, Melbourne, Australia
Ioan Alfred Letia, TU Cluj Napoca, Romania
Isabel Praca, Istit. Superior do Porto, Portugal
Agostinho Rosa, Technical University Lisbon, Portugal

Simulation and Optimization
José António Oliveira, Universidade do Minho, Campus de Gualtar, Braga, Portugal
Sameiro Carvalho, Universidade do Minho, Campus de Gualtar, Braga, Portugal

High Performance Large Scale and Hybrid Computing
Jan Broeckhove, University of Antwerp, Antwerp, Belgium
Simon See, Oracle, Singapore
Pierre Siron, ONERA, Toulouse, France
Jingjing Wang, SUNY Binghamton University, New York, USA

Simulation in Education and Graphics Visualization
Track Chair
Ana Luisa Ramos, University of Aveiro, Aveiro, Portugal

IPC
Marco Roccetti, University of Bologna, Italy
Ranjit Singh, UB Patient Safety Research Center, University of Buffalo, Buffalo NY USA

Simulation in Environment, Ecology, Biology and Medicine
Eduardo Ayesa, CEIT, San Sebastian, Spain
Joel Colloc, LITIS, Le Havre, France
Laurent Perochon, VetaGrSc-Sup, Lempdes, France
Filipe Pinto, Polytechnic Institute of Leiria, Portugal

Analytical and Numerical Modelling Techniques
Clemens Heitzinger, University of Vienna, Vienna, Austria

Web Based Simulation
Manuel Alfonseca, Universidad Autonoma de Madrid, Spain
Ammar Al-Khani, Nordphysics, Espoo, Finland
Yan Luo, NIST, Gaithersburg, USA
Jose Machado, University of Minho, Braga, Portugal
Mara Nikolaidou, University of Athens, Greece
Vaclav Snasel, VSB Technical University of Ostrava, Czech Republic

Cosmological Simulation
José Manuel Feliz-Teixeira, University of Porto, Porto, Portugal
Philippe Geril, ETI Bvba, Ostend, Belgium
INTERNATIONAL PROGRAMME COMMITTEE

Simulation-based evaluation in Living Labs

Track Chairs
Stefan Hellfeld, FZI - House of Living Labs, Karlsruhe, Germany
Benjamin Herd, King’s College, London, United Kingdom
Stefanie Betz, Karlsruhe Institute of Technology, Karlsruhe, Germany
Thomas Karle, PROMATIS software GmbH, Ettlingen, Germany
Andreas Oberweis, Karlsruhe Institute of Technology, Karlsruhe, Germany
Polina Häfner, Karlsruhe Institute of Technology, Karlsruhe, Germany

Simulation and Modelling for humanitarian/emergency operations
Hella Abidi, Ild Institute for Logistics and Service Management at FOM, Essen, Germany

Intelligent Systems
Track Chair
Leandros Maglaras, De Montfort University, Leicester, United Kingdom
Ying He, De Montfort University, Leicester, United Kingdom
José Machado, Universidade do Minho, Braga, Portugal
Antonio Abélha, Universidade do Minho, Braga, Portugal
Manuel Filipe Santos, Universidade do Minho, Guimarães, Portugal
Victor Alves, Universidade do Minho, Guimarães, Portugal

Real-time GPS Simulation and Service Applications
Matthias Klumpp, FOM University of Applied Sciences, Essen, Germany

Simulation with Petri Nets
Track Chair
François Siewe, De Montfort University, Leicester, United Kingdom
Pascal Berruet, Université Bretagne Sud, Lorient, France
Olivier Grunder, UTBM, Belfort, France
Stefano Marrone, Seconda Universita degli Studi di Napoli, Naples, Italy
Alexandre Nketsa, LAAS-CNRS, Toulouse, France
Jean-Claude Pascal, LAAS-CNRS, Toulouse, France

Bond Graphs Simulation
Jesus Felez, Univ. Politecnica de Madrid, Spain
Aziz Naaman, DIAM-IUSPIM, Marseille, France
Andre Tavernier, BioSim, Brussels, Belgium

DEVs
Fernando Tricas, Universidad de Zaragoza, Spain

Fluid Flow Simulation
H.A. Nour Eldin, University of Wuppertal, Germany
Markus Fiedler, Blekinge Institute of Technology, Sweden

Emergency Risk Management Simulation
Véronique Baudin, LAAS-CNRS, Université de Toulouse, Toulouse, France
Kamil Drira, LAAS-CNRS, Université de Toulouse, Toulouse, France
Konstantinos Krytopoulos, University of South Australia, Australia
Sakravarthi Ramanathan, Muthayammal Engineering College, India
Gurdev Singh, UB Patient Safety Research Center, State University of New York at Buffalo, Buffalo NY USA
Lode Vermeersch, Delcredere, Brussels, Belgium
EUROPEAN SIMULATION AND MODELLING CONFERENCE 2015
© 2015 EUROYSIS-ETI

Responsibility for the accuracy of all statements in each peer-referenced paper rests solely with the author(s). Statements are not necessarily representative of nor endorsed by the European Simulation Society. Permission is granted to photocopy portions of the publication for personal use and for the use of students providing credit is given to the conference and publication. Permission does not extend to other types of reproduction nor to copying for incorporation into commercial advertising nor for any other profit-making purpose. Other publications are encouraged to include 300- to 500-word abstracts or excerpts from any paper contained in this book, provided credits are given to the author and the conference.

All author contact information provided in this Proceedings falls under the European Privacy Law and may not be used in any form, written or electronic, without the written permission of the author and the publisher.

All articles published in these Proceedings have been peer reviewed
A CIP Catalogue record for this book is available from the Royal Library of Belgium under nr.12620

For permission to publish a complete paper write EUROYSIS, c/o Philippe Geril, ETI Executive Director, Greenbridge Science Park, Ghent University – Ostend Campus, Wetenschapspark 1, Plassendale 1, B-8400 Ostend, Belgium.

EUROYSIS is a Division of ETI Bvba, The European Technology Institute, Torhoutseesteenweg 162, Box 4, B-8400 Ostend, Belgium

Printed in Belgium by Reproduct NV, Ghent, Belgium
Cover Design by Grafisch Bedrijf Lammaing, Ostend, Belgium

ESM® is a European registered trademark of the European Technology Institute under nr: 002433290

EUROYSIS-ETI Publication

EAN: 9789077381908
Preface

It is our privilege and pleasure to welcome you to the 29th European Simulation and Modelling Conference - ESM'2015 held in association with the Faculty of Technology, De Montfort University, in Leicester. As in previous editions, this conference promotes the knowledge exchange in the fields of Simulation and Modelling, allowing participants to share their experiences in an open forum providing an enriching environment and stimulating debates.

We have more than seventy participants from 25 countries with presentations grouped in 21 main themes such as Complex Intermodal Transport Systems; Virtual Production Environments; Intelligent Data Simulation; Simulation Tools; Meta Resource Models; to mention a few. The Keynote Speaker, Professor Cyrille Bertelle, will address an important topic in simulation and modelling with his special tribute talk to the late John Forbes Nash covering Local Nash Equilibrium in Social Networks.

Leicester, pronounced Lester, is a city in the East Midlands of England, and the county town of Leicestershire. It lies on the River Soar and at the edge of the National Forest. Recently, it made the headlines with the news of discovering the remains of King Richard III. The remains have now been reburied in the City Cathedral with an elaborate memorial. Leicester, from its early settlement as Ratae, was a Roman military outpost in a region inhabited by the Celtic Corieltauvi tribe. Ancient Roman pavements and baths remain can still be seen. It also played an important role in post-Roman Britain history. It was mentioned in the Domesday Book as "Ledecestre". The Earls of Leicester also played an important role in medieval Britain politics, from which Simon De Montfort has a distinctive recognition. Modern Leicester continues to be a vibrant hub for business, culture and learning.

We would like to welcome all participants and to thank all authors for sharing their knowledge and experience. Our thanks also go to all members of the Program Committee for the reviewing work that was key to maintaining the high scientific quality of ESM'2015. Our gratitude also goes to the Keynote Speaker.

We are grateful to the members of the Organizing Committee for their help in preparing the conference. A special thanks to Philippe Geril from EUROSIIS, whose continued dedication and hard work, as the conference organizer, has enabled us to maintain the standard expected for ESM2015. Finally, we would like to express our gratitude to DMU for its support and to the hosting venue for a job well done.

Last but not least, we would like to wish you all a fruitful and productive experience at the conference and an enjoyable and enriching stay in Leicester.

Aladdin Ayesh and Marwan Al-Akaidi
(General and Program Chairman)
CONTENTS

Preface ........................................................................................................................................ IX
Scientific Programme .................................................................................................................. 1
Author Listing ............................................................................................................................ 463

SIMULATION FUNDAMENTALS

Big Data Reduction  
William Conley ....................................................................................................................... 5

Traversing Two and Twelve Dimensional Space  
William Conley .......................................................................................................................... 10

Modelling and Analysis of Hybrid Physical Systems Using Bond Graph  
and Dynamic Causality  
J. Madrigal Salas and R.F. Ngwompo .......................................................................................... 16

Hybrid Synchronization in the DCB Based on Uncoordinated Checkpoints  
Flávio Marques Migowski Carvalho and Braulio Adriano de Mello ........................................... 22

META-RESOURCE MODELS

A Meta-Model for Consistent & Automatic Simulation Model Selection  
Sangeeth Saagar Ponnumasy, Vincent Albert and Patrice Thebault ........................................ 31

A Resource Model for the Rule-Based Dynamic Business Process  
Modelling and Simulation  
Olegas Vasilecas, Diana Kalibatiene, Audrius Rima, Ilze Birzniece  
and Peteris Rudzajs ..................................................................................................................... 36

Flat vs Nested Firm Real-time Transactions: an Analysis of their Success Ratio  
Mourad Kaddes, Laurent Amanton, Majed Abdouli, Mohalilou Aleiyouka  
and Alexandre Berred .................................................................................................................. 42

Discrete Event Simulation to Reduce the Effect of Uncertainties on  
Project Planning  
Parminder Singh Kang, Adam Aboutaleb, Tim Embley, James Glenn,  
Colin Adams and Alistair Duffy .................................................................................................... 47
CONTENTS

INTELLIGENT DATA SIMULATION

Metaheuristics Applied to the Autonomous Movement of Intelligent Agents
Dâmaris da Silva Bento, Bábara da Costa Rodrigues, Juliano Costa Penha Alves,
Bruno Luan de Sousa, Angelo Batista Neves Júnior, Lindomar Matos de Paulo
and Alex F.V. Machado.................................................................57

Predicting 2 Way Football Results by Means of Data Mining
João Gomes, Filipe Portela, Manuel Felipe Santos, António Abelha
and José Machado.................................................................64

PETRI NET SIMULATION

Quantitative Analysis of the Publish/Subscribe Paradigm in the Context
of Web Service Resources with Timed Colored Petri Nets
Valentín Valero, Hermenegilda Macià and Gregorio Díaz ......................73

Deriving the Performance Indices in Product-Form Stochastic Petri Nets:
Open Problems and Simulation
Simonetta Balsamo, Andrea Marin and Ivan Stojic ..........................81

SIMULATION TOOLS

ASSISI Playground: A Simulation Tool for Collective Bio-Hybrid System
Research
Pedro Mariano, Rob Mills, Luís Correia, Damjan Miklić and Stjepan Bogdan...91

Synthesis of a Three Stages Servovalve by AMESIM
Costin Ina, Nicolae Vasiliu, Constantin Călinoiu, Daniela Vasiliu
and Marius Daniel Bontos..........................................................99

SOFTWARE ENGINEERING SIMULATION

New Codesign Solutions for Modelling and Partitioning of Probabilistic
Reconfigurable Embedded Software
Ines Ghribi, Riadh Ben Abdallah, Mohamed Khalgui and Marco Platzner .......107

Modeling Issues when using Simulation to test The Performance of
Mathematical Programming Models under Stochastic Conditions
Anne-Laure Ladier, Allen G. Greenwood and Gülgün Alpan ..................117

Function Slices: A Model to Extract Parallelism from Sequential
Applications
Nils Potthoff, Carsten Gremzow, Christoph Brandau and Dietmar Tutsch .......122
CONTENTS

DEVS

Optimizing Distributed DEVS Simulations with Partitioning and Hidden
Markov Model Learning Methods
Christopher Herbez, Eric Ramat and Gauthier Quesnel .........................133

A New Approach to Modeling Dynamic Structure Systems
Oumar Maïga, Hamzat Olanrewaju Aliyu and Mamadou Kaba Traoré ..........141

A Framework for Discrete Events Systems Enactment
Hamzat Olanrewaju Aliyu, Oumar Maïga and Mamadou Kaba Traoré ..........149

HARDWARE OPTIMIZATION

Signal Processing at Different Mental Tasks Execution for Brain-Computer
Interface
Svetla Radeva and Dimitar Radev ..................................................159

A Based-Optimization Scheduling Approach for Multi-Processor Platform
with DVFS Facilities
Hamza Chnitier, Fethi Jarray, Mohamed Khalgui and Maryline Chetto ........164

Efficient Techniques for Test Generation and Validation of Reconfigurable
Hardware Systems
Asma Ben Ahmed, Olfa Mosbah and Mohamed Khalgui ..........................172

HEALTH CARE SIMULATION

An Agent-Based Approach to reveal the Effects of Age-Related Contact
Patterns on Epidemic Spread
Dietmar Neubacher, Nikolaus Furian and Siegfried Vößner .......................185

Towards a Framework for Holistic Analysis of Healthcare Systems
Ignace Djitog, Hamzat Olanrewayu Aliyu and Mamadou Kaba Traoré ..........193

Simulating the Impact of Optimized Dispatching Strategies for Patient
Transits
Nikolaus Furian, Michael O'Sullivan, Cameron Walker, Simon Rieboz,
Dietmar Neubacher and Siegfried Vößner .........................................197

ENVIRONMENTAL SIMULATION

Comparing Neural Networks and Pseudo Inverse Matrix to Flood Forecasting
of the Amazon River
Marcio Menezes, Marly Costa and Cicero Costa Filho ............................205
CONTENTS

Performance Evaluation of Discontinuity Tracking in Vector Fields for Avalanche Simulation
Ángel Rodríguez-Cerro, Miguel Lozano, Ignacio García-Fernández
and Rafael J. Martínez-Durá ................................................................. 211

WATER DISTRIBUTION SYSTEMS

Model Calibration and Hydraulic Optimization of the Water Distribution System of the Upper Silesian Waterworks
Jan Studzinski, Andrzej Ziółkowski and Dorota Tomasiuk .................. 219

A Simulation Study of Minimizing Water Cost in Multisource Water Distribution System
Rafał Brodziak, Jędrzej Bylka, Andrzej Urbaniak
and Przemysław Zakrzewski ............................................................... 225

Modelling and Optimization of Communal Sewerage Systems
Grazyna Petriczek and Jan Studzinski ................................................. 229

ENERGY CONSUMPTION SIMULATION

Optimal Load Sharing Strategy in System of Multiple Source Based on Model Control Predictive
Mouhamadou Falilou Ndiaye, Pape Alioune Ndiaye, François Guerin
and Dimitri Lefebvre ............................................................................ 237

Wireless Sensor Network Live Circle Simulation Tools To Balance Energy Consumption in the Network Nodes
Aleksejs Jurenoks and Leonids Nocickis .............................................. 245

Simulation of Change Power Demand in Micro-Grid Supplied by Mix of Sources including Aggregate of Small Scale Hydropower Plants
Eugeniusz M. Sroczan ........................................................................ 251

Real-time Task Reconfiguration in Energy-Harvesting based Multiprocessor Systems
Wiem Housseyni, Olfa Mosbah, Mohamed Khalgui and Maryline Chetto .... 254

Application of Intelligent Control Algorithms for Thermal Comfort and Energy Saving in the Computer Laboratory
Mariusz Nowak and Andrzej Urbaniak ................................................. 265
ENGINEERING SIMULATION

CFD Analysis on Characteristics of CANDU6 Ventilation/Cooling Systems
S. Nicolici, C.T. Mateescu, R. Arsene, I. Prisecaru and D. Dupleac ...............273

Modelling of the Vertical Apparent Mass at the Seat Surface and Backrest
Naser Nayayseh..................................................................................278

Mathematical Analysis on Pitch and Roll Variations Based Hit Probability
Prediction for a Combat Vehicle Design
Hwan Il Kang and Hyun Soo Kim..........................................................284

Analysis of Design Parameters Based on Bullet Trajectories and Hit
Probability
Hwan Il Kang, Sang Yeong Choi and Hyun Soo Kim...............................289

AUTOMOTIVE SIMULATION

Driving Simulator Development Phase II – Building and Controlling
Scenarios through the Motion Algorithm
Alexandra Tuca, Valerian Crotorescu and Thomas Brancmeier ..............297

Modeling and Formal Verification of Reconfigurable Vehicular Platoons
Emna Guerfala, Mohamed Khalgui, Anis Koubâa and Oussema Karoui ......302

GRAPHICS AND MEDIA NETWORK SIMULATION

Visualisation of Physical Simulation using Javascript Physics Engine
with X3DOM
Olaf Berndt and Linda Huber ..................................................................315

Video Replication over DASH in Distributed Multimedia Systems
Haifa Harrouch, Mourad Kaddes, Claude Duvallet, Majed Abdouli
and Rafik Bouaziz...............................................................................318

Study of the Disturbance Spread in an Interaction Network
Haifa Rabai, Rodolphe Charrier and Cyrille Bertelle.................................323

VIRTUAL REALITY SIMULATION

3D Surgery Simulation Software Development Kit
Anton Ivaschenko, Alexandr Kolsanov, Aikush Nazaryan
and Andrey Kuzmin.............................................................................333
CONTENTS

The Need of Simulation in Queuing Models for Port Logistics
Pasquale Legato and Rina M. Mazza .................................................................418

Impact of Different Input Data Sets on the Sorting System Performance of a Parcel Transshipment Terminal
Daniel Diekmann, Uwe Clausen and Ina Dormuth...........................................423

Pickup and Delivery Selection with a Fixed Vehicle Cost
Katrien Ramaekers, Gerrit K. Janssens, Tabitha Maes and An Caris..............428

Using Agent Based Simulation to Investigate Daily Order Variation of a B2B Fresh Food Supplier
Ross Clement, Parminder Kang, Alistair Duffy and Ashley Hopwell .............433

LATE PAPERS

Modeling and Simulation of Tawaf and Sa’Yee: A Survey of Recent Work in the Field
Omar Kurdi, Mike Stannett and Daniela M. Romano ......................................441

Procedural Modelling Techniques to Configure Scenes in a Serious Game Based Driving Simulator
Pedro Rosa and Rosaldo J.F. Rossetti .................................................................448

An Extension to the FEEDB Multimodal Database of Facial Expressions and Emotions
Luis Marco-Giménez, Miguel Arevalillo-Herráez, Aladdin Ayesh and Mariusz Szocho .................................................................455
SCIENTIFIC PROGRAMME
SIMULATION FUNDAMENTALS
BIG DATA REDUCTION

William Conley
Austin E. Cofrin School of Business
480B Wood Hall
University of Wisconsin-Green Bay
Green Bay, Wisconsin 54311-7001 U.S.A.
Conleyw@uwgb.edu

KEYWORDS
Statistical correlation, complex systems, data spreadsheets, multi stage Monte Carlo optimization

ABSTRACT
Another name for statistics is “data reduction.” The computer world of our new age with its increasingly available large data sets (spreadsheets) from the internet (and other sources) can be a bit overwhelming at times to researchers, scientists, engineers, politicians and other decision makers. The new multivariate correlation statistic CTSP can be used to reduce these large spreadsheets of data to perhaps show whether or not correlation exists between the variables represented by the large data sets under consideration. The CTSP statistic is used here to discover correlation between ten variables that are represented in a 10x89 array of data. The theory and technique are presented.

INTRODUCTION
The multi stage Monte Carlo Optimization (MSMCO) simulation shortest route algorithm is used to calculate CTSP on an example involving n=89 lines of ten columns of data in Table 1 (the columns represent ten variables). This will help to determine if correlation exists between the ten variables. This approach relies on the fact that relatively shorter shortest routes through data indicate that the variables that produced the data are correlated.

Table 1

<table>
<thead>
<tr>
<th></th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
<th>x5</th>
<th>x6</th>
<th>x7</th>
<th>x8</th>
<th>x9</th>
<th>x10</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>20.0</td>
<td>4.0</td>
<td>53.0</td>
<td>75.0</td>
<td>83.0</td>
<td>37.0</td>
<td>6.0</td>
<td>14.0</td>
<td>26.0</td>
<td>46.5</td>
</tr>
<tr>
<td>2</td>
<td>29.0</td>
<td>43.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>100.0</td>
<td>27.0</td>
<td>43.0</td>
<td>11.0</td>
<td>97.0</td>
</tr>
<tr>
<td>3</td>
<td>62.0</td>
<td>28.0</td>
<td>26.0</td>
<td>5.0</td>
<td>13.0</td>
<td>86.0</td>
<td>79.0</td>
<td>19.0</td>
<td>32.0</td>
<td>24.3</td>
</tr>
<tr>
<td>4</td>
<td>17.0</td>
<td>21.0</td>
<td>78.0</td>
<td>67.0</td>
<td>6.0</td>
<td>15.0</td>
<td>37.0</td>
<td>87.0</td>
<td>79.0</td>
<td>16.5</td>
</tr>
<tr>
<td>5</td>
<td>6.0</td>
<td>63.0</td>
<td>3.0</td>
<td>79.0</td>
<td>84.0</td>
<td>42.0</td>
<td>20.0</td>
<td>55.0</td>
<td>41.0</td>
<td>41.0</td>
</tr>
<tr>
<td>6</td>
<td>67.0</td>
<td>26.0</td>
<td>78.0</td>
<td>22.0</td>
<td>66.0</td>
<td>45.0</td>
<td>96.0</td>
<td>67.0</td>
<td>60.0</td>
<td>39.5</td>
</tr>
<tr>
<td>7</td>
<td>70.0</td>
<td>92.0</td>
<td>28.0</td>
<td>72.0</td>
<td>91.0</td>
<td>15.0</td>
<td>6.0</td>
<td>7.0</td>
<td>91.0</td>
<td>58.5</td>
</tr>
<tr>
<td>8</td>
<td>66.0</td>
<td>100.0</td>
<td>2.0</td>
<td>58.0</td>
<td>56.0</td>
<td>41.0</td>
<td>60.0</td>
<td>88.0</td>
<td>95.0</td>
<td>32.1</td>
</tr>
<tr>
<td>9</td>
<td>94.0</td>
<td>21.0</td>
<td>14.0</td>
<td>100.0</td>
<td>76.0</td>
<td>15.0</td>
<td>33.0</td>
<td>63.0</td>
<td>78.0</td>
<td>39.5</td>
</tr>
<tr>
<td>10</td>
<td>5.0</td>
<td>70.0</td>
<td>52.0</td>
<td>1.0</td>
<td>17.0</td>
<td>10.0</td>
<td>8.0</td>
<td>63.0</td>
<td>95.0</td>
<td>13.1</td>
</tr>
<tr>
<td>11</td>
<td>4.0</td>
<td>4.0</td>
<td>96.0</td>
<td>10.0</td>
<td>98.0</td>
<td>98.0</td>
<td>17.0</td>
<td>60.0</td>
<td>100.0</td>
<td>56.6</td>
</tr>
<tr>
<td>12</td>
<td>70.0</td>
<td>32.0</td>
<td>85.0</td>
<td>13.0</td>
<td>75.0</td>
<td>100.0</td>
<td>48.0</td>
<td>90.0</td>
<td>100.0</td>
<td>48.0</td>
</tr>
<tr>
<td>13</td>
<td>20.0</td>
<td>37.0</td>
<td>36.0</td>
<td>94.0</td>
<td>33.0</td>
<td>88.0</td>
<td>23.0</td>
<td>79.0</td>
<td>52.0</td>
<td>23.9</td>
</tr>
<tr>
<td>14</td>
<td>21.0</td>
<td>71.0</td>
<td>27.0</td>
<td>49.0</td>
<td>49.0</td>
<td>82.0</td>
<td>46.0</td>
<td>68.0</td>
<td>96.0</td>
<td>25.6</td>
</tr>
<tr>
<td>15</td>
<td>67.0</td>
<td>58.0</td>
<td>59.0</td>
<td>41.0</td>
<td>35.0</td>
<td>17.0</td>
<td>100.0</td>
<td>50.0</td>
<td>98.0</td>
<td>3.5</td>
</tr>
<tr>
<td>16</td>
<td>31.0</td>
<td>37.0</td>
<td>46.0</td>
<td>41.0</td>
<td>100.0</td>
<td>1.0</td>
<td>30.0</td>
<td>69.0</td>
<td>33.0</td>
<td>59.4</td>
</tr>
<tr>
<td>17</td>
<td>2.0</td>
<td>25.0</td>
<td>26.0</td>
<td>32.0</td>
<td>63.0</td>
<td>64.0</td>
<td>26.0</td>
<td>100.0</td>
<td>68.0</td>
<td>26.4</td>
</tr>
<tr>
<td>18</td>
<td>100.0</td>
<td>100.0</td>
<td>8.0</td>
<td>80.0</td>
<td>8.0</td>
<td>9.0</td>
<td>90.0</td>
<td>33.0</td>
<td>18.0</td>
<td>36.6</td>
</tr>
<tr>
<td>19</td>
<td>19.0</td>
<td>59.0</td>
<td>72.0</td>
<td>6.0</td>
<td>39.0</td>
<td>86.0</td>
<td>52.0</td>
<td>67.0</td>
<td>32.0</td>
<td>25.5</td>
</tr>
<tr>
<td>20</td>
<td>9.0</td>
<td>82.0</td>
<td>93.0</td>
<td>29.0</td>
<td>63.0</td>
<td>38.0</td>
<td>83.0</td>
<td>45.0</td>
<td>52.0</td>
<td>36.2</td>
</tr>
<tr>
<td>21</td>
<td>9.0</td>
<td>0.0</td>
<td>24.0</td>
<td>36.0</td>
<td>94.0</td>
<td>38.0</td>
<td>12.0</td>
<td>47.0</td>
<td>65.0</td>
<td>41.8</td>
</tr>
<tr>
<td>22</td>
<td>72.0</td>
<td>53.0</td>
<td>91.0</td>
<td>64.0</td>
<td>61.0</td>
<td>86.0</td>
<td>72.0</td>
<td>72.0</td>
<td>100.0</td>
<td>43.3</td>
</tr>
<tr>
<td>23</td>
<td>55.0</td>
<td>64.0</td>
<td>92.0</td>
<td>86.0</td>
<td>50.0</td>
<td>9.0</td>
<td>27.0</td>
<td>80.0</td>
<td>25.0</td>
<td>37.5</td>
</tr>
<tr>
<td>24</td>
<td>58.0</td>
<td>14.0</td>
<td>88.0</td>
<td>87.0</td>
<td>100.0</td>
<td>15.0</td>
<td>15.0</td>
<td>78.0</td>
<td>17.0</td>
<td>68.7</td>
</tr>
<tr>
<td>25</td>
<td>33.0</td>
<td>44.0</td>
<td>69.0</td>
<td>14.0</td>
<td>25.0</td>
<td>25.0</td>
<td>29.0</td>
<td>54.0</td>
<td>60.0</td>
<td>13.7</td>
</tr>
<tr>
<td>26</td>
<td>85.0</td>
<td>70.0</td>
<td>76.0</td>
<td>33.0</td>
<td>53.0</td>
<td>54.0</td>
<td>57.0</td>
<td>66.0</td>
<td>89.0</td>
<td>31.0</td>
</tr>
<tr>
<td>27</td>
<td>47.0</td>
<td>4.0</td>
<td>24.0</td>
<td>100.0</td>
<td>33.0</td>
<td>8.0</td>
<td>61.0</td>
<td>84.0</td>
<td>58.0</td>
<td>21.9</td>
</tr>
<tr>
<td>28</td>
<td>13.0</td>
<td>77.0</td>
<td>26.0</td>
<td>96.0</td>
<td>5.0</td>
<td>33.0</td>
<td>49.0</td>
<td>98.0</td>
<td>41.0</td>
<td>22.3</td>
</tr>
<tr>
<td>29</td>
<td>99.0</td>
<td>19.0</td>
<td>62.0</td>
<td>92.0</td>
<td>44.0</td>
<td>63.0</td>
<td>84.0</td>
<td>40.0</td>
<td>10.0</td>
<td>41.3</td>
</tr>
<tr>
<td>30</td>
<td>12.0</td>
<td>85.0</td>
<td>77.0</td>
<td>17.0</td>
<td>66.0</td>
<td>56.0</td>
<td>58.0</td>
<td>50.0</td>
<td>93.0</td>
<td>32.3</td>
</tr>
<tr>
<td>31</td>
<td>1.0</td>
<td>8.0</td>
<td>33.0</td>
<td>22.0</td>
<td>69.0</td>
<td>0.0</td>
<td>13.0</td>
<td>76.0</td>
<td>20.0</td>
<td>32.8</td>
</tr>
<tr>
<td>32</td>
<td>66.0</td>
<td>2.0</td>
<td>48.0</td>
<td>56.0</td>
<td>70.0</td>
<td>31.0</td>
<td>84.0</td>
<td>13.0</td>
<td>56.0</td>
<td>48.2</td>
</tr>
</tbody>
</table>
THE CTSP IDEA IN TWO DIMENSIONS

Consider the two sets of n=52 x, y points graphed in Figure 1. The 52 points on the border are following a pattern. Therefore, that data came from two x and y variables that are highly correlated (in a piecewise linear fashion in this case).

Notice also the n=52 x y points graphed inside the border in Figure 1. These points are not following a pattern (so they are fairly random) indicating that those two x and y variables are not correlated. Observe also that a shortest route connecting all of their points in a closed loop is much longer than the shortest route connecting the n=52 points on the border. These facts can be exploited in the CTSP correlation statistic to test for correlation. The
The author has tested hundreds of data sets of three through twelve dimensions (even though the geometry is hard to picture). Closeness can be defined algebraically in higher dimensions. The tests run on the three through nine variables correlation studies were done in the years 2002 through 2014. The tests on the ten, eleven and twelve dimensional spreadsheets of data were conducted in 2015.

Figure 1: The 52 points on the border are from x and y variables that are highly correlated. However, the 52 points inside the border came from x and y variables that are not correlated; hence, a longer shortest route.

Now look at the n=56 points on the border in Figure 2. Its shortest route is much shorter than the shortest route connecting the n=56 points inside the border of Figure 2. Therefore, the x y variables that produced the border points are correlated, but the x and y variables that gave rise to the points inside the border are not correlated.

Let us look at a ten variable example.

Figure 2: The 56 points on the border have a short shortest route and their variables are correlated. The 56 points inside the border have a longer shortest route and no correlation is present.

AN ENGINEERING STUDY: TEN VARIABLES SPREADSHEET

The company engineers have been collecting data on 89 trials involving 10 variables that they think might be correlated. The data is in Table 1.

They want to use the new CTSP correlation statistic because they think the relationship (if it exists) may not be linear and CTSP is more general than the standard big R used on linear correlations.

Therefore, they use the multi stage Monte Carlo optimization (MSMCO) algorithm to find a short route (on a complete tour) through ten dimensional space for the n=89 points. A generalization of the Pythagorean formula based distance measure is adjusted for ten variables to measure distances. The shortest route has distance d=6542.813 for the n=89 points.
29 32 67 50 58 2 65 15 73 45
87 78 52 76 34 1 83 5 55 60
59 28 71 33 43 3 82 72 27 57
4 66 25 10 61 53 51 7 89 14
46 42 36 11 12 68 6 88 39 35
19 20 30 29 22 26 64 49 38 81
41 40 18 37 44 3 54 77 75 69
13 47 86 17 21 31 16 24 70 62
9 85 23 48 56 74 84 63 80 29

Reading left to right and top to bottom, the route goes from point 29 to point 32 to point 67 . . . and on to point 80 and back to point 29.

Now five sets of random data in the same ranges produced short routes (again using MSMCO) of $d_2 = 6953.843$, $d_3 = 6891.376$, $d_4 = 6815.492$, $d_5 = 7097.790$ and $d_6 = 6890.657$. Now taking the quotients of the twenty pairs of short route distances from the random data we get a good range for CTSP of .96 to 1.041 if the null hypothesis $H_0$: of no correlation is true.

\[
\begin{align*}
6953.843/6891.376 &= 1.009 \\
6953.843/6815.492 &= 1.020 \\
6953.843/7097.790 &= .979 \\
6891.376/6953.843 &= .991 \\
6815.492/6953.843 &= .980 \\
7097.790/6953.843 &= 1.021 \\
6890.657/6953.843 &= .991 \\
6891.376/6815.492 &= 1.011 \\
6891.376/7097.790 &= .971 \\
6891.376/6890.657 &= 1.000 \\
6815.492/6891.376 &= .989 \\
7097.790/6891.376 &= 1.030 \\
6890.657/6891.376 &= 1.000 \\
6815.492/7097.790 &= .960 \\
6815.492/6890.657 &= .989 \\
7097.790/6815.492 &= 1.041 \\
6890.657/6815.492 &= 1.011 \\
7097.790/6890.657 &= 1.030 \\
6890.657/7097.790 &= .971
\end{align*}
\]

Now the median of the five short route distances from the random data sets is 6891.376.

The statistic $CTSP = d/\text{median} = 6542.813/6891.376 = .949$ in this case which is below the range of .960 to 1.041. Therefore, the null hypothesis ($H_0$) of no correlation between the variables can be rejected and $H_A$ of a correlation between the variables accepted. It turns out that the equation $x_{10} = 21.25 + .001x_5 + .07(x_1 + x_2 + x_3 + x_4) + .0019x_6 + .1444x_5 + .002x_7 + .116x_7 + .002x_8 - .208x_8 + .002x_9 - .3x_9$ fits the data quite well, confirming the engineer's idea that variables $x_1$ through $x_9$ were driving the production yield variable of $x_{10}$.

**MULTI-STAGE MONTE CARLO OPTIMIZATION**

The statistical simulation based Multi Stage Monte Carlo optimization (MSMCO) solution technique was used here to find shortest routes (complete tours) through $k=10$ dimensional points with the goal of determining whether or not the data came from variables that were correlated. The MSMCO simulation rode $k=10$ dimensional “rectangles” of ever repositioning and decreasing in size to the minimum solution region of the feasible solution space, looking for shortest routes. Please see the left hand side of Figure 3 for a partial geometric and statistical representation of a two dimensional projection of the $k=10$ dimensional approach.

![Figure 3: k dimensional “rectangles” looking for the minimum solution while m dimensional “ellipsoids” close in on a solution to a maximum production solution using multi stage Monte Carlo optimization](image)

However, MSMCO is a fairly general purpose optimization technique that can be used to also maximize functions like production, profit, chemical yields, sales, customer satisfaction and pain relief, etc. Please see the right hand side of Figure 3 for a two dimensional projection of ellipsoids closing in on the maximum solution to an optimization problem.

The element of risk is certainly presented in any multi stage Monte Carlo simulation. However, with the increase in speed capacity and affordability of today’s modern computers and some careful geometric and probabilistic calculations to control the sampling and
closings in on the optimal solution region the “risk” can be managed so that the probability of finding the true optimal or a useful approximate solution is quite high.

The traditional approaches to correlation analysis are available in (Anderson 2003), (Black 2014), (Anderson, Sweeney and Williams 1999), (Hayter 2002) and (Keller and Warrack 2003). The CTSP formula is a new entry into the area of correlation analysis.

CONCLUSION

Presented here was a hypothetical research problem where a large spread sheet of data (89 lines of 10 columns representing 10 variables) was “reduced” in a sense to the conclusion that the ten variables were correlated. The correlation turned out to be nonlinear in this case. This ten variable problem is larger than the CTSP examples previously presented. The new (developed this century) CTSP correlation statistic (Conley 2007), (Conley 2013), (Conley 2006) in conjunction with the multi stage Monte Carlo optimization (MSMCO) approach (Conley 1981), (Conley 1991), (Szarkowicz 1995), (Wong 1996), (Conley 2013b), and (Conley 2008) to solving problems was used here. The problem was hypothetical. However, the CTSP multivariate correlation statistic is very real and can help researchers and decision makers of all kinds to “navigate” through “BIG DATA” to find out what useful information may be hiding in these higher dimensional problems.

This is a new era (as our computer age goes forward) allowing one to take shortest routes through spreadsheets of data to find information that may be “hidden” there. Then decision makers can take the appropriate action based on these results. The MSMCO simulation approach searches for optimal solutions rather than estimating probability as so many simulations do.

REFERENCES


BIOGRAPHY

WILLIAM CONLEY received a B.A. in mathematics (with honors) from Albion College in 1970, an M.A. in mathematics from Western Michigan University in 1971, M.Sc. in statistics in 1973 and a Ph.D. in mathematics - computer statistics from the University of Windsor in 1976. He has taught mathematics, statistics, and computer programming in universities for over 30 years. He is currently a professor emeritus of Business Administration and Statistics at the University of Wisconsin at Green Bay. The developer of multi stage Monte Carlo optimization and the CTSP multivariate correlation statistics, he is the author of five books and more than 200 publications world- wide. He is a member of the American Chemical Society, a fellow in the Institution of Electronic and Telecommunication Engineers and a senior member of the Society for Computer Simulation.
TRAVERSING TWO AND TWELVE DIMENSIONAL SPACE

William Conley
Austin E. Cofrin School of Business
480B Wood Hall
University of Wisconsin-Green Bay
Green Bay, Wisconsin 54311-7001 U.S.A.
Conleyw@uwgb.edu

KEYWORDS
Complex systems, multivariate statistical analysis, optimization, shortest routes

ABSTRACT

Simulation techniques can be used to solve or approximate the solution to large scale shortest route problems. Generally these have been restricted to two and three dimensional problems for surface and airfreight deliveries in the past. After a brief review of these ideas in two or more dimensions, twelve dimensional data are studied by finding shortest routes through them and exploiting those shortest route results to prove whether the data came from variables that are correlated or not. The new CTSP correlation coefficient statistic is used for the analysis. Shortest route technology can deliver data analysis in our computer age. This can help to reduce large data sets to meaningful information which is paramount in our 21st century which can overwhelm us with data.

INTRODUCTION

Presented here is a drawing (please see Figure 1) with two sets of n=28 points on an x y plane. The 28 points on the border came from x and y variables that are highly correlated (following a pattern). Therefore, the shortest route connecting them is much shorter (about 30% shorter) than a shortest route connecting the n=28 points that are inside the border. Those points are not following a pattern; therefore, the x and y variables they represent are not correlated. This fact has been tested hundreds of times by the author and it holds true in higher dimensional space (more variables) where it is very difficult to represent it pictorially. Some three, eight and nine dimensional examples are in (Conley 2006) and (Conley 2007).

Featured here are shortest route calculations on twelve dimensional data sets (please see Figure 2 for a partial representation of finding these shortest routes). The first four come from random data and their shortest routes (through twelve dimensional space) are much longer than the shortest routes through the last two twelve dimensional data sets. Those facts, along with the CTSP statistical calculations will show that those two sets came from variables that are highly correlated.

SOME RANDOM DATASETS

Researchers think that it is possible that the n=80 lines of 12 columns of numbers in Data Set One (in Table 1) have come from 12 variables (represented by the columns) that are correlated. Therefore, to test this idea, four sets of random data in similar ranges are created (n=80 lines for each of them) with a random number generator.

Then the multi stage Monte Carlo optimization (MSMCO) statistical simulation technique is used to find shortest routes through these four sets of random data. The distance metric used is the generalization of the Pythagorean Theorem distance formula adjusted for twelve dimensional space.

The distances are $d_1=7807.560$, $d_2=7809.254$, $d_3=7577.155$ and $d_4=7548.751$. Therefore, the median of these four distances is $(7807.560 + 7577.155)/2=7692.3575$. Our CTSP statistic is the shortest route from the real data divided by the median of the shortest routes from the random data. Calculated CTSP values that are much less than, one indicates correlation between the variables in the test data set.
Figure 1: The 28 XY Points in the Middle Are Not Correlated. However, XY points on the Boarders are Correlated.

Figure 2: A Partial Geometric and Statistical Representation of the MSMCO Shortest Route Algorithm Tracking and Finding the Six Shortest Routes, Four From Random Data and Two from Correlated Data, Presented Here.

<table>
<thead>
<tr>
<th></th>
<th>X_1</th>
<th>X_2</th>
<th>X_3</th>
<th>X_4</th>
<th>X_5</th>
<th>X_6</th>
<th>X_7</th>
<th>X_8</th>
<th>X_9</th>
<th>X_{10}</th>
<th>X_{11}</th>
<th>X_{12}</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>38.8</td>
<td>44.0</td>
<td>9.5</td>
<td>4.0</td>
<td>15.0</td>
<td>58.3</td>
<td>97.0</td>
<td>76.0</td>
<td>2.0</td>
<td>15.1</td>
<td>17.0</td>
<td>40.5</td>
</tr>
<tr>
<td>2</td>
<td>5.7</td>
<td>6.0</td>
<td>58.0</td>
<td>100.0</td>
<td>16.0</td>
<td>61.3</td>
<td>78.0</td>
<td>70.0</td>
<td>36.0</td>
<td>80.8</td>
<td>98.0</td>
<td>5.8</td>
</tr>
<tr>
<td>3</td>
<td>79.2</td>
<td>92.0</td>
<td>50.0</td>
<td>91.0</td>
<td>9.0</td>
<td>32.0</td>
<td>23.3</td>
<td>26.0</td>
<td>47.0</td>
<td>3.8</td>
<td>4.0</td>
<td>83.3</td>
</tr>
<tr>
<td>4</td>
<td>43.1</td>
<td>49.0</td>
<td>32.0</td>
<td>43.0</td>
<td>21.0</td>
<td>62.9</td>
<td>100.0</td>
<td>6.0</td>
<td>83.0</td>
<td>14.2</td>
<td>16.0</td>
<td>45.0</td>
</tr>
<tr>
<td>5</td>
<td>75.9</td>
<td>88.0</td>
<td>30.5</td>
<td>50.0</td>
<td>11.0</td>
<td>55.3</td>
<td>81.0</td>
<td>57.0</td>
<td>28.0</td>
<td>5.6</td>
<td>6.0</td>
<td>79.7</td>
</tr>
<tr>
<td>6</td>
<td>25.1</td>
<td>28.0</td>
<td>92.0</td>
<td>85.0</td>
<td>99.0</td>
<td>25.0</td>
<td>18.0</td>
<td>2.0</td>
<td>55.0</td>
<td>57.0</td>
<td>68.0</td>
<td>26.0</td>
</tr>
<tr>
<td>7</td>
<td>61.7</td>
<td>71.0</td>
<td>31.5</td>
<td>1.0</td>
<td>62.0</td>
<td>69.9</td>
<td>30.3</td>
<td>80.0</td>
<td>100.0</td>
<td>0.0</td>
<td>0.0</td>
<td>64.6</td>
</tr>
<tr>
<td>8</td>
<td>19.9</td>
<td>22.0</td>
<td>66.0</td>
<td>45.0</td>
<td>87.0</td>
<td>38.3</td>
<td>40.3</td>
<td>47.0</td>
<td>28.0</td>
<td>28.5</td>
<td>33.0</td>
<td>20.6</td>
</tr>
<tr>
<td>9</td>
<td>14.6</td>
<td>16.0</td>
<td>56.5</td>
<td>78.0</td>
<td>35.0</td>
<td>74.3</td>
<td>80.3</td>
<td>93.0</td>
<td>50.0</td>
<td>11.7</td>
<td>13.0</td>
<td>15.1</td>
</tr>
<tr>
<td>10</td>
<td>68.4</td>
<td>79.0</td>
<td>50.5</td>
<td>65.0</td>
<td>36.0</td>
<td>55.6</td>
<td>97.0</td>
<td>44.0</td>
<td>26.0</td>
<td>57.8</td>
<td>69.0</td>
<td>71.8</td>
</tr>
<tr>
<td>11</td>
<td>14.6</td>
<td>16.0</td>
<td>67.0</td>
<td>34.0</td>
<td>100.0</td>
<td>72.3</td>
<td>86.3</td>
<td>84.0</td>
<td>47.0</td>
<td>64.2</td>
<td>77.0</td>
<td>15.1</td>
</tr>
<tr>
<td>12</td>
<td>2.0</td>
<td>2.0</td>
<td>79.0</td>
<td>79.0</td>
<td>79.0</td>
<td>31.0</td>
<td>17.0</td>
<td>2.0</td>
<td>74.0</td>
<td>3.8</td>
<td>4.0</td>
<td>2.0</td>
</tr>
<tr>
<td>13</td>
<td>83.4</td>
<td>97.0</td>
<td>73.5</td>
<td>52.0</td>
<td>95.0</td>
<td>83.9</td>
<td>61.3</td>
<td>91.0</td>
<td>100.0</td>
<td>21.0</td>
<td>24.0</td>
<td>87.7</td>
</tr>
<tr>
<td>14</td>
<td>27.7</td>
<td>31.0</td>
<td>53.5</td>
<td>73.0</td>
<td>34.0</td>
<td>73.3</td>
<td>60.0</td>
<td>70.0</td>
<td>90.0</td>
<td>49.7</td>
<td>59.0</td>
<td>28.7</td>
</tr>
<tr>
<td>15</td>
<td>71.7</td>
<td>83.0</td>
<td>61.5</td>
<td>23.0</td>
<td>100.0</td>
<td>60.3</td>
<td>2.0</td>
<td>100.0</td>
<td>79.0</td>
<td>63.4</td>
<td>76.0</td>
<td>75.3</td>
</tr>
<tr>
<td>16</td>
<td>1.0</td>
<td>1.0</td>
<td>43.5</td>
<td>79.0</td>
<td>8.0</td>
<td>57.3</td>
<td>35.3</td>
<td>95.0</td>
<td>42.0</td>
<td>32.6</td>
<td>38.0</td>
<td>1.0</td>
</tr>
<tr>
<td>17</td>
<td>37.1</td>
<td>42.0</td>
<td>37.0</td>
<td>51.0</td>
<td>23.0</td>
<td>70.6</td>
<td>87.0</td>
<td>25.0</td>
<td>100.0</td>
<td>9.9</td>
<td>11.0</td>
<td>38.7</td>
</tr>
<tr>
<td>18</td>
<td>21.6</td>
<td>24.0</td>
<td>6.0</td>
<td>10.0</td>
<td>2.0</td>
<td>62.6</td>
<td>50.0</td>
<td>84.0</td>
<td>54.0</td>
<td>42.4</td>
<td>50.0</td>
<td>22.4</td>
</tr>
<tr>
<td>19</td>
<td>44.8</td>
<td>51.0</td>
<td>49.5</td>
<td>45.0</td>
<td>54.0</td>
<td>71.3</td>
<td>78.0</td>
<td>90.0</td>
<td>46.0</td>
<td>53.7</td>
<td>64.0</td>
<td>46.8</td>
</tr>
<tr>
<td>20</td>
<td>57.5</td>
<td>66.0</td>
<td>85.5</td>
<td>80.0</td>
<td>91.0</td>
<td>50.0</td>
<td>46.3</td>
<td>75.0</td>
<td>29.0</td>
<td>10.8</td>
<td>12.0</td>
<td>60.2</td>
</tr>
<tr>
<td>21</td>
<td>3.8</td>
<td>4.0</td>
<td>29.5</td>
<td>7.0</td>
<td>52.0</td>
<td>56.3</td>
<td>48.0</td>
<td>94.0</td>
<td>27.0</td>
<td>78.5</td>
<td>95.0</td>
<td>3.9</td>
</tr>
<tr>
<td>22</td>
<td>85.9</td>
<td>100.0</td>
<td>18.5</td>
<td>7.0</td>
<td>30.0</td>
<td>56.6</td>
<td>25.0</td>
<td>69.0</td>
<td>76.0</td>
<td>47.3</td>
<td>56.0</td>
<td>90.4</td>
</tr>
<tr>
<td>23</td>
<td>9.3</td>
<td>10.0</td>
<td>28.5</td>
<td>54.0</td>
<td>3.0</td>
<td>51.3</td>
<td>99.0</td>
<td>31.0</td>
<td>24.0</td>
<td>59.4</td>
<td>71.0</td>
<td>9.5</td>
</tr>
<tr>
<td>24</td>
<td>13.7</td>
<td>15.0</td>
<td>63.5</td>
<td>61.0</td>
<td>66.0</td>
<td>15.0</td>
<td>43.0</td>
<td>11.0</td>
<td>30.0</td>
<td>32.6</td>
<td>38.0</td>
<td>14.1</td>
</tr>
<tr>
<td>25</td>
<td>85.9</td>
<td>100.0</td>
<td>49.0</td>
<td>7.0</td>
<td>28.0</td>
<td>41.3</td>
<td>35.3</td>
<td>42.0</td>
<td>47.0</td>
<td>59.4</td>
<td>71.0</td>
<td>90.4</td>
</tr>
<tr>
<td>26</td>
<td>49.0</td>
<td>56.0</td>
<td>38.5</td>
<td>60.0</td>
<td>17.0</td>
<td>48.3</td>
<td>26.0</td>
<td>36.0</td>
<td>83.0</td>
<td>35.9</td>
<td>42.0</td>
<td>51.3</td>
</tr>
<tr>
<td>27</td>
<td>84.2</td>
<td>98.0</td>
<td>22.5</td>
<td>42.0</td>
<td>3.0</td>
<td>33.0</td>
<td>25.3</td>
<td>13.0</td>
<td>61.0</td>
<td>31.8</td>
<td>37.0</td>
<td>88.6</td>
</tr>
<tr>
<td>28</td>
<td>12.8</td>
<td>14.0</td>
<td>93.0</td>
<td>100.0</td>
<td>86.0</td>
<td>49.0</td>
<td>19.3</td>
<td>28.0</td>
<td>100.0</td>
<td>82.4</td>
<td>100.0</td>
<td>13.2</td>
</tr>
<tr>
<td>29</td>
<td>48.2</td>
<td>55.0</td>
<td>62.5</td>
<td>31.0</td>
<td>94.0</td>
<td>61.6</td>
<td>71.3</td>
<td>48.0</td>
<td>66.0</td>
<td>76.1</td>
<td>92.0</td>
<td>50.4</td>
</tr>
<tr>
<td>30</td>
<td>85.9</td>
<td>100.0</td>
<td>62.0</td>
<td>83.0</td>
<td>41.0</td>
<td>42.0</td>
<td>12.3</td>
<td>77.0</td>
<td>37.0</td>
<td>67.3</td>
<td>81.0</td>
<td>90.4</td>
</tr>
<tr>
<td>31</td>
<td>22.5</td>
<td>25.0</td>
<td>74.0</td>
<td>86.0</td>
<td>62.0</td>
<td>36.6</td>
<td>38.0</td>
<td>59.0</td>
<td>13.0</td>
<td>64.2</td>
<td>77.0</td>
<td>23.3</td>
</tr>
<tr>
<td>32</td>
<td>2.0</td>
<td>2.0</td>
<td>23.5</td>
<td>14.0</td>
<td>33.0</td>
<td>77.9</td>
<td>98.0</td>
<td>41.0</td>
<td>95.0</td>
<td>9.9</td>
<td>11.0</td>
<td>2.0</td>
</tr>
</tbody>
</table>
The shortest route calculation (on the real data) using the multi stage Monte Carlo optimization (MSMCO) simulation on a desk top computer produced a shortest route solution of total distance $d=5686.161$.

\[ \begin{array}{cccccccc}
33 & 69.2 & 80.0 & 5.0 & 4.0 & 6.0 & 65.3 & 64.3 & 41.0 \\
34 & 78.4 & 91.0 & 78.5 & 64.0 & 93.0 & 71.6 & 74.0 & 100.0 \\
35 & 64.2 & 74.0 & 90.5 & 89.0 & 92.0 & 54.9 & 71.0 & 75.0 \\
36 & 47.3 & 54.0 & 64.5 & 29.0 & 100.0 & 49.0 & 93.0 & 22.0 \\
37 & 80.9 & 93.0 & 45.5 & 63.0 & 28.0 & 69.9 & 66.3 & 76.0 \\
38 & 85.9 & 100.0 & 22.0 & 13.0 & 31.0 & 11.3 & 23.3 & 1.0 \\
39 & 85.9 & 100.0 & 52.5 & 43.0 & 62.0 & 37.3 & 73.3 & 9.0 \\
40 & 15.5 & 17.0 & 5.0 & 1.0 & 9.0 & 51.3 & 63.3 & 49.0 \\
41 & 75.1 & 87.0 & 56.5 & 79.0 & 34.0 & 38.3 & 13.3 & 61.0 \\
42 & 47.3 & 54.0 & 29.0 & 14.0 & 44.0 & 41.0 & 53.3 & 3.0 \\
43 & 12.8 & 14.0 & 68.0 & 79.0 & 57.0 & 45.6 & 57.3 & 10.0 \\
44 & 76.7 & 89.0 & 96.0 & 100.0 & 92.0 & 21.0 & 18.3 & 45.0 \\
45 & 65.9 & 75.0 & 90.5 & 88.0 & 93.0 & 39.0 & 89.3 & 28.0 \\
46 & 79.2 & 92.0 & 77.5 & 71.0 & 84.0 & 72.9 & 76.3 & 100.0 \\
47 & 19.9 & 22.0 & 54.5 & 100.0 & 9.0 & 61.6 & 53.3 & 57.0 \\
48 & 22.5 & 25.0 & 10.0 & 18.0 & 2.0 & 39.3 & 75.3 & 43.0 \\
49 & 32.0 & 36.0 & 64.0 & 32.0 & 96.0 & 50.9 & 47.3 & 76.0 \\
50 & 85.1 & 99.0 & 31.0 & 49.0 & 13.0 & 20.6 & 32.3 & 15.0 \\
51 & 60.8 & 70.0 & 80.0 & 49.0 & 69.0 & 40.0 & 48.3 & 13.0 \\
52 & 32.8 & 37.0 & 43.0 & 71.0 & 15.0 & 66.3 & 86.3 & 22.0 \\
53 & 67.6 & 78.0 & 18.5 & 19.0 & 18.0 & 52.9 & 100.3 & 4.0 \\
54 & 64.2 & 74.0 & 67.0 & 48.0 & 86.0 & 81.3 & 96.3 & 54.0 \\
55 & 84.2 & 98.0 & 41.0 & 54.0 & 28.0 & 66.6 & 48.0 & 99.0 \\
56 & 2.0 & 2.0 & 66.5 & 100.0 & 33.0 & 31.6 & 12.3 & 72.0 \\
57 & 55.8 & 64.0 & 71.0 & 61.0 & 81.0 & 76.9 & 33.3 & 100.0 \\
58 & 63.4 & 73.0 & 40.0 & 30.0 & 50.0 & 65.6 & 100.0 & 41.0 \\
59 & 79.2 & 92.0 & 88.0 & 76.0 & 100.0 & 25.3 & 17.9 & 1.0 \\
60 & 36.3 & 41.0 & 53.0 & 48.0 & 56.0 & 60.3 & 74.3 & 20.0 \\
61 & 75.9 & 88.0 & 45.0 & 23.0 & 67.0 & 46.0 & 81.0 & 38.0 \\
62 & 50.7 & 58.0 & 65.0 & 87.0 & 43.0 & 45.3 & 21.0 & 88.0 \\
63 & 28.5 & 32.0 & 49.5 & 53.0 & 46.0 & 69.6 & 39.9 & 75.0 \\
64 & 38.8 & 44.0 & 20.5 & 9.0 & 32.0 & 36.3 & 76.0 & 13.0 \\
65 & 3.8 & 4.0 & 46.0 & 91.0 & 1.0 & 75.9 & 56.3 & 82.0 \\
66 & 43.1 & 49.0 & 39.5 & 69.0 & 10.0 & 10.0 & 1.0 & 21.0 \\
67 & 43.9 & 50.0 & 52.5 & 66.0 & 39.0 & 46.0 & 55.3 & 37.0 \\
68 & 38.0 & 43.0 & 87.0 & 74.0 & 99.0 & 75.6 & 87.0 & 41.0 \\
69 & 85.9 & 100.0 & 96.5 & 94.0 & 77.0 & 38.0 & 16.3 & 95.0 \\
70 & 63.4 & 73.0 & 67.0 & 57.0 & 77.0 & 36.0 & 12.3 & 60.0 \\
71 & 53.3 & 61.0 & 40.0 & 75.0 & 5.0 & 37.3 & 18.0 & 67.0 \\
72 & 42.2 & 48.0 & 56.5 & 67.0 & 46.0 & 86.2 & 67.0 & 98.0 \\
73 & 16.4 & 18.0 & 85.0 & 82.0 & 88.0 & 29.3 & 8.0 & 7.0 \\
74 & 5.7 & 6.0 & 37.5 & 0.0 & 75.0 & 52.6 & 100.0 & 37.0 \\
75 & 26.8 & 30.0 & 79.0 & 71.0 & 87.0 & 28.0 & 55.3 & 15.0 \\
76 & 26.8 & 30.0 & 79.0 & 71.0 & 87.0 & 32.6 & 33.3 & 22.0 \\
77 & 44.8 & 51.0 & 43.0 & 76.0 & 10.0 & 41.3 & 36.3 & 74.0 \\
78 & 15.5 & 17.0 & 50.0 & 32.0 & 68.0 & 64.3 & 96.3 & 92.0 \\
79 & 28.5 & 32.0 & 19.5 & 9.0 & 30.0 & 58.9 & 3.0 & 74.0 \\
80 & & & & & & & & \\
\end{array}\]

Reading the route from left to right top to bottom it went from twelve dimensional point 27 (line number 27 in the data) to point 3 (line 3) to point 59 (line 59) and so on down to point 38 (line 38) and then back to point 27 (line 27) traveling a distance of 5686.161 units.

Therefore, CTSP = $d/\text{median} = 5686.161/7692.3575 = .7392$. Now taking our quotients from the shortest routes from the random data we get an estimate of the sampling distribution in the test of hypotheses.

Ho: the twelve variables are not correlated.

\( H_\alpha: \text{Ho is false.} \)

The quotients are:

\[ 7807.560/7809.254 = .9998 \]
\[ 7807.560/7757.255 = 1.0304 \]
7807.560/.7548.751 = 1.0343
7809.254/.7807.560 = 1.0002
7577.155/.7807.560 = .9705
7548.751/.7807.560 = .9669
7809.254/.7577.155 = 1.0306
789.254/.7548.751 = 1.0345
7577.155/.7809.254 = .9703
7548.751/.7809.254 = .9667
7577.155/.7548.751 = 1.0038
7548.751/.7577.155 = .9963
or a range of .9667 to 1.0345. Therefore, the vast majority of the sampling distribution of CTSP must be in this range if there is no correlation. However, our CTSP value is .7392 so the null hypothesis Ho can be rejected.

Therefore, the conclusion is that the 12 variables in data set one are highly correlated.

It turns out that the piecewise linear and nonlinear functions

\[ x_{10} = x_{11} \]

fit the data very well.

**A SECOND CORRELATION QUESTION**

Another data set (please see Data Set Two in Table 2) is being studied to see if the variables represented by the numbers in the columns of the n=80 lines of this new data are variables that are correlated or not. As the range (0-100) and the number of lines of data (n=80) is the same as in the previous example we can use the same range of .9667 to 1.0345 as the region (around one) that will hold the vast majority of the sampling distribution of CTSP if the null hypothesis Ho is true. The researchers want to test the null hypothesis Ho: no correlation between the variables versus HA: a correlation between the variables.

Therefore, the MSMCQ statistical optimization simulation algorithm is used again to find a shortest route through this new spreadsheet of data in twelve dimensional space. The route has a length of d=2051.326 total distance for a complete tour from point

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
<th>x5</th>
<th>x6</th>
<th>x7</th>
<th>x8</th>
<th>x9</th>
<th>x10</th>
<th>x11</th>
<th>x12</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>2</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>3</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>4</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>5</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>6</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>7</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>8</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>9</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>10</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>11</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>12</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>13</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>14</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>15</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>16</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>17</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>18</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>19</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>20</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>21</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>22</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>23</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>24</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>25</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>26</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>27</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>28</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>29</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
<tr>
<td>30</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>100.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
<td>50.0</td>
</tr>
</tbody>
</table>

and back to point 13 reading left to right and from top to bottom in the printout.
Therefore, \( \text{CTSP} = \frac{2051.326}{7692.3575} = .2667 \) which is so far away from one in relative terms that if \( Ho \) of no correlation is true, we have observed a result which is unbelievable. Therefore \( Ho \) is false and the variables are highly correlated.

It turns out that the relation \( \sum_{i=1}^{12}(x_i-\overline{x})^2 = 2500 \) fits the data quite well. One could call it a “spheroid” in twelve dimensional space. A spheroid in twelve variables is sort of like a “round” World Cup football (or soccer ball) in twelve dimensions.

**STATISTICAL OPTIMIZATION**

Every large scale optimization problem presents some kind of multivariate formula (call it a statistic if you wish) where the goal is to find the maximum (production, horsepower, thrust, profit, etc.) or the minimum (cost, pollution, error in solving equations, etc.) Therefore, that formula has a sampling distribution of all of the feasible solutions. Please see (Conley 1980) and (Conley 1981) for many graphs of the exact sampling distribution of several multivariate optimization problems.

Therefore, computer simulation can be used very effectively to cross these sampling distributions to the optimal (or a useful approximation). The author prefers
repeated “random” simulations” controlled by the trail of better and better answers, in ever repositioning and decreasing in size geometric shapes (Please see Figure 2 for a partial statistical and geometric illustration). Further examples of these are in (Wong 1996), (Szarkowicz 1995), and (Conley 2013).

The name used here for this simulation optimization approach to problem solving is statistical optimization or Multi Stage Monte Carlo Optimization (MSMCO) (Conley, 1981), (Conley 1991) and Conley (2007).

CONCLUSION

The authors (Black 2014), (Anderson, Sweeney, Williams, 1999), (Keller, et al 2003), (Klibanoff, et al 2006), (Levine, et al 2002), (McClave 2001), (Anderson 2003), and (Hayter 2002) presented the standard linear correlation analysis (for two variables r) and its multivariate generalization big R (Anderson 2003) under certain regularity conditions. Presented here is a multivariate alternative that works on linear, nonlinear and even nonfunctional multivariate correlation problems. It also has the advantage of not squaring any error terms (which is often required in the standard least squares approach to correlation analysis).

Specifically, two large scale multivariate correlation problems involving n=80 lines of 12 dimensional data were presented. Both were found to have variables that were correlated using the multi stage Monte Carlo statistical optimization simulation approach to find a shortest routes connecting the data points in twelve dimensional space.

This does not mean that least squares analysis is no longer useful. However, the big area of multivariate nonlinear correlation analysis needs some new solution approaches. Researchers can now follow the shortest route to information (“correlation analysis”) in addition to delivering products and services at lower cost on a shortest route.

REFERENCES


BIOGRAPHY

WILLIAM CONLEY received a B.A. in mathematics (with honors) from Albion College in 1970, an M.A. in mathematics from Western Michigan University in 1971, an M.Sc. in statistics in 1973 and a Ph.D. in mathematics - computer statistics from the University of Windsor in 1976. He has taught mathematics, statistics, and computer programming in universities for over 30 years. He is currently a professor emeritus of Business Administration and Statistics at the University of Wisconsin at Green Bay. The developer of multi stage Monte Carlo optimization and the CTSP multivariate correlation statistics, he is the author of five books and more than 200 publications world- wide. He is a member of the American Chemical Society, a fellow in the Institution of Electronic and Telecommunication Engineers and a senior member of the Society for Computer Simulation. Career highlights include presentation of two papers at National Aeronautics and Space Administration (NASA) conferences in Houston, Texas and Washington, D.C.
MODELLING AND ANALYSIS OF HYBRID PHYSICAL SYSTEMS USING BOND GRAPH AND DYNAMIC CAUSALITY

J Madrigal Salas and R F Ngwombo
Department of Mechanical Engineering
University of Bath
BA2 7AY, Bath,
UK
E-mail: jams21@bath.ac.uk, r.f.ngwombo@bath.ac.uk

KEYWORDS
Hybrid systems, switched systems, bond graph, dynamic causality, implicit model.

ABSTRACT

Analysis of systems are often done from a Linear Time Invariant (LTI) approach. This approach does not represent the continuous and discontinuous behaviour of a hybrid or switched system on a single model. As a result, hybrid systems are often represented as a sequence of LTI equations depending on the configuration of switches or modes of operation.

Bond graph are an established modelling method, in which there are several methods to represent the commuting elements. This paper aims to set a standard notation for hybrid bond graph in order to derive a unique representation that encompasses all modes of operations and from which multi-modes analysis can be undertaken. Switching elements are modelled using controlled junctions and this leads to the use of dynamic causality for mathematical model generation and model analysis. From the junction structure of the dynamic causal bond graph representing the hybrid system, a mixed-Boolean implicit unique model is derived for the system enabling the determination of valid operational modes.

INTRODUCTION

The analysis of physical systems is usually based on mathematical equations. These equations describe the system behaviour by decomposing the interactions of the elements found on the systems. There are some representations that are usually avoided on mathematical expressions. This is due to discontinuities on the system or interaction between different types of systems (mechanical, electrical, hydraulics, thermal, magnetic, to name some) that could create instabilities or undesired behaviours during simulation. A different approach is necessary to analyse these behaviours, namely, bond graph methodology. Bond graph is used due to its unified approach to multidisciplinary systems and its ability to represent physical behaviour.

Bond graph methodology provides a unified representation of the different elements present on any system, whatever their nature or complexity. This leads to an unified representation of the state equation in any physical domain.

For hybrid bond graphs, controlled junctions and dynamic causality are used in this work. This is due to hybrid bond graphs incorporating both continuous and discontinuous behaviour of the system. There are a number of proposed approaches to modelling the commuting elements on the system. None has reached a common usage, nevertheless, there is a particular representation that is of interest for this analysis. This is the controlled junction which allows obtaining the (usually) missing physical behaviours not taken into consideration by other representations.

This representation was introduced by Mosterman and Biswas (Mosterman and Biswas 1997), and later formalized by Margetts et al (Margetts et al. 2013) introducing the change of causality during commutation. This change on the causality has been avoided in order to keep the system on the same order for all the available operational modes. This is usually done by adding a parasitic or ‘causality’ resistance with the purpose of facilitating simulations ((Buisson 1993a), (Buisson 1993b), (Junco et al. 2008)), usually leading to a complex model or generating a model that differs from the physical system.

With the inclusion of controlled switched junctions and dynamic causality, the use of Boolean parameters is necessary in order to have a simplified notation of the operational modes available on the system and depending on the state of the switches which may be ON or OFF. This paper is organized as follows: section 2 recalls the analysis of linear systems using bond graph, in section 3 an introduction to hybrid bond graph analysis is presented. The state equations and valid operational modes are obtained in section 4 and 5. Some examples are given in section 6.

STANDARD BOND GRAPH ANALYSIS

Consider a linear time-invariant system represented in bond graph. Its junction structure and key vectors are shown in Fig. 1.

In Fig. 1, the source field contains sources that can be either flow or effort \( (S_c, S_f) \), the storage field contains capacitive or inductive elements \( (C, I) \), the resistive field contains the resistive elements \( (R) \), and the junction structure contains the common effort or flow junctions, transformers and gyrators \( (0, 1, TF, GY) \). \( x \) denotes the state variables and is composed of energy variables; \( z \) is the co-energy variables vector; \( U \) is the system input;
and finally $D_{in}$ and $D_{out}$ are a mixture of $e$ and $f$ showing the energy exchange between the resistive field and the junction structure. $x_i$ and $z_d$ denote variables associated with elements in integral causality, while $x_d$ and $z_d$ are associated with elements in derivative causality.

**Fig. 1** Junction structure of a Bond Graph

The relationships defining the storage and resistive fields are

$$z_i = F_z x_i$$
$$z_d = F_d x_d$$
$$D_{out} = LD_{in}$$ (1)

And the junction structure matrix is given by equation (2)

$$\begin{bmatrix}
\dot{x}_i \\
\dot{z}_d \\
D_{out}
\end{bmatrix} =
\begin{bmatrix}
S_{11} & S_{12} & S_{13} & S_{14} \\
-S_{12}^T & 0 & 0 & 0 \\
-S_{13}^T & 0 & S_{33} & S_{34}
\end{bmatrix}
\begin{bmatrix}
z_i \\
\dot{x}_d \\
D_{in} \\
U
\end{bmatrix}$$

(2)

From row 3 of equation (2) an expression for $D_{out}$ is obtained:

$$D_{out} = -S_{13}^T z_i + S_{33} D_{in} + S_{34} U$$

(3)

By substituting (1) in (3) results:

$$D_{in} = \left( L \left( I - S_{33} L^{-1} \right) \right)^{-1} \left( -S_{13}^T z_i + S_{34} U \right)$$

Therefore, the term $D_{in}$ can be eliminated from the equations. From row 1 of equation (2) $\dot{x}$ is:

$$\dot{x}_i = (S_{11} - S_{13} S_{33}^T) z_i + S_{12} \dot{x}_d$$
$$+ (S_{14} + S_{13} S_{34} S_{34}^T) U$$

where $M = \left( L \left( I - S_{33} L^{-1} \right) \right)^{-1}$. Obtaining a state equation of the form:

$$\dot{x} = Ax + Bu$$

where

$$A = E^{-1} \left( S_{11} - S_{13} S_{33}^T \right) F_i$$
$$B = E^{-1} \left( S_{14} + S_{13} S_{34} S_{34}^T \right)$$
$$E = I + S_{12} F_i^{-1} S_{34}^T F_i$$

This analysis is only valid for linear time invariant systems. In order to use bond graph for hybrid physical systems, controlled junctions and dynamic causality need to be represented. This is done by introducing controlled junctions and dynamic causality representation into the structure of the system. By doing this, new matrices appear and some of the junction structure matrices change due to the effects created by the controlled junction behaviour.

**HYBRID BOND GRAPH ANALYSIS**

Analysis of hybrid bond graph models requires the representation of the commuting elements behaviour and their effects on the systems dynamic. In this case, controlled junctions represent the dynamic behaviour of the commutation. The use of this representation introduces a dynamic causality to the analysis of the systems. Dynamic causality allows a more physical related behaviour of the elements during commutation to be obtained. This is done by representing the change of the power transfer between elements when a commuting element creates a new causal path associated with a controlled junction changing its state. By introducing these modelling representations to the general junction structure of the system, a matrix based on Boolean parameters can be obtained. This Boolean matrix contains the representation of the commuting behaviour of the controlled junctions with a 1 (resp. 0) Boolean variable associated to the ON (resp. OFF) state.

**Controlled Junctions**

Switching elements (dis)connect or (de)activate part of the model. In order to analyse this behaviour there is the need of representing it on the model as well. Controlled junctions are used for their representation of the structural discontinuities where the elements connect and disconnect and break the path of power flow. Controlled junctions can be represented as X1 or X0 (Fig. 2).

A Boolean parameter $\lambda$ is associated to the controlled junction element, which is considered as a 2-port element, although it could be generalised to n-port. This $\lambda$ parameter is used to indicate the ON or OFF state of the element.
Fig. 2 Bond graph representation of controlled junctions X1 and X0

When controlled junctions are used in the modelling of a system, a dynamic change of causality appears. This is due to the fact that every time that a controlled junction change its state the power flow is inhibited or it is transferred to another element. X1 or X0 junctions behave as a normal 1 or 0 junctions when ON, and a source of zero flow or zero effort (respectively) when OFF. Therefore, a controlled 1 junction is used to inhibit flow and a controlled 0 junction to inhibit effort.

Dynamic Causality

On linear system analysis, causality assignment in bond graph is assigned using the Sequential Causality Assignment Procedure (SCAP) (Karnopp et al. 2006). For hybrid models there are some alternatives, which are focused on having a static causality assignment for an efficient simulation. Nevertheless, this causality assignment does not represent the system’s behaviour in all the modes. In this case, dynamic causality is unavoidable when using controlled junctions without parasitic resistive elements.

The causality assignment starts with a reference operational mode defined with a maximum number of storage elements in integral causality and controlled junctions in ON state. This procedure is called dynamic sequential causality assignment procedure (DSCAP) of a hybrid bond graph and is summarised in (Margetts et al. 2003).

If a causal conflict occurs or a source element is connected to the ground element, that operational mode is not valid.

Dynamic causality creates a new representation on the junction structure matrix for the elements affected by the behaviour of the commutating elements.

Boolean Matrix

This Boolean matrix displays the commutation of the switching elements ((Muller 1954), (Staehler 1952)). Also, this matrix allows a simplified version of the junction structure matrix, in which all the available operational modes are represented.

As the junction structure matrix is in the form $EX = Ax + Bu$, the matrix $E$ contains the $\lambda$ parameters associated to the commutating elements. This matrix represents all the available combinations of commutating elements interacting in the system.

It must be noted that the matrices $A$ and $B$ also contains $\lambda$ parameters.

The analysis of these matrices is done by substituting the values of the $\lambda$ parameters with either 0 or 1 depending on the state of the controlled junction. When a commutation occurs, some lines and rows are removed in order to eliminate redundancies in the matrices.

Junction Structure

Usually the interactions between elements on a junction structure matrix consist of ones and zeros. For hybrid bond graph this is still valid, nevertheless, in order to properly represent all the interactions between the different elements on all available operational modes, a new general representation of the junction structure is introduced.

Due to the reference operational mode having the maximum number of storage elements with integral causality in addition with the use of switched junctions, it is assured that there are no storage elements with static derivative causality.

In Fig. 3, it can be seen that there is a differentiation between the elements affected by the dynamic causality and those that remains with static causality.

In order to make a distinction between the storage elements with static and storage elements with dynamic causality, the symbol ~ is added on top of the notation of the elements with dynamic causality. The symbol ~ on top of the elements represent the complement for the commutating elements.

Fig. 3: Switched junction structure

In this representation, the ground connection is introduced to the source field. This due to the interaction between a switched element and ground that could create a non-valid mode. The non-valid mode could represent a short circuit, an anchorage, to name a few.

As can be seen, the fields affected by dynamic causality are displayed in order to set a differentiation with the non-dynamic fields. The junction structure matrix for this can be written as:

$$
\begin{bmatrix}
1 & 0 & 0 & \cdots & 0 & 0 \\
0 & \lambda_{x_1} & 0 & \cdots & 0 & 0 \\
0 & 0 & \lambda_{x_2} & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & \lambda_{x_m} & 0 \\
0 & 0 & 0 & \cdots & 0 & \lambda_{x_1}
\end{bmatrix}
\begin{bmatrix}
\dot{x}_1 \\
\dot{x}_2 \\
\vdots \\
\dot{x}_n \\
\end{bmatrix}
= 
\begin{bmatrix}
\dot{z}_1 \\
\dot{z}_2 \\
\vdots \\
\dot{z}_m
\end{bmatrix}
$$

(4)
This matrix is now in the form \( \Lambda \mathbf{x} = S \mathbf{x} \). Where \( \Lambda_{22} \) and \( \Lambda_{34} \) are the matrices for a state of the switches and the complement state at commutation respectively. This is to have a clear distinction between the configurations of the storage elements affected by dynamic causality. This is due to the fact that some causality assignment can create a non-defined preferred integral or derivative causality on storage elements. Also \( \Lambda_{44} \) and \( \Lambda_{44} \) show the configuration for the resistive elements affected by dynamic causality for a state of switches and their complement.

Here \( \tilde{D}_{out} \) generates a row and column full of 0s. This is due to the absence of flow or effort on the incoming port when there is a commutation on the controlled junction.

Thus, \( \Lambda_{44} \) and \( \tilde{D}_{out} \) can be eliminated from the matrix.

\[
\begin{bmatrix}
I & 0 & 0 & 0 \\
0 & \Lambda_{22} & 0 & 0 \\
0 & 0 & \tilde{\Lambda}_{22} & 0 \\
0 & 0 & 0 & \tilde{D}_{out}
\end{bmatrix}
\begin{bmatrix}
\dot{\mathbf{x}}_1 \\
\dot{\mathbf{x}}_2 \\
\dot{\mathbf{x}}_3 \\
\dot{\mathbf{x}}_4
\end{bmatrix} =
\begin{bmatrix}
\frac{S_{11}}{S_{11}} & \frac{S_{12}}{S_{12}} & \frac{S_{13}}{S_{13}} & \frac{S_{14}}{S_{14}} \\
\frac{-S_{21}^T}{S_{21}^T} & \frac{S_{22}}{S_{22}} & \frac{S_{23}}{S_{23}} & \frac{S_{24}}{S_{24}} \\
\frac{-S_{31}^T}{S_{31}^T} & \frac{S_{32}}{S_{32}} & \frac{S_{33}}{S_{33}} & \frac{S_{34}}{S_{34}} \\
\frac{-S_{41}^T}{S_{41}^T} & \frac{-S_{42}^T}{-S_{42}^T} & \frac{S_{43}}{S_{43}} & \frac{S_{44}}{S_{44}}
\end{bmatrix}
\begin{bmatrix}
\mathbf{z}_1 \\
\mathbf{z}_2 \\
\mathbf{z}_3 \\
\mathbf{z}_4
\end{bmatrix}
\]

In order to simplify the expressions, \( D_{out} \) and \( \tilde{D}_{out} \), \( \tilde{D}_{in} \) and \( \tilde{D}_{in} \) will be considered as a single element ( \( D_{out} = \frac{D_{out}}{D_{out}} \), \( D_{in} = \frac{D_{in}}{D_{in}} \)). In addition, their respective matrices will be considered as a single one.

This matrix is noted as \( \Lambda_{34} = \begin{bmatrix} I & 0 \\ 0 & \Lambda_{44} \end{bmatrix} \). These changes are represented in the next simplified matrix

\[
\begin{bmatrix}
I & 0 & 0 & 0 \\
0 & \Lambda_{32} & 0 & 0 \\
0 & 0 & \Lambda_{32} & 0 \\
0 & 0 & 0 & \Lambda_{44}
\end{bmatrix}
\begin{bmatrix}
\dot{\mathbf{x}}_1 \\
\dot{\mathbf{x}}_2 \\
\dot{\mathbf{x}}_3 \\
\dot{\mathbf{x}}_4
\end{bmatrix} =
\begin{bmatrix}
\frac{S_{11}}{S_{11}} & \frac{S_{12}}{S_{12}} & \frac{S_{13}}{S_{13}} & \frac{S_{14}}{S_{14}} \\
\frac{-S_{21}^T}{S_{21}^T} & \frac{S_{22}}{S_{22}} & \frac{S_{23}}{S_{23}} & \frac{S_{24}}{S_{24}} \\
\frac{-S_{31}^T}{S_{31}^T} & \frac{S_{32}}{S_{32}} & \frac{S_{33}}{S_{33}} & \frac{S_{34}}{S_{34}} \\
\frac{-S_{41}^T}{S_{41}^T} & \frac{-S_{42}^T}{-S_{42}^T} & \frac{S_{43}}{S_{43}} & \frac{S_{44}}{S_{44}}
\end{bmatrix}
\begin{bmatrix}
\mathbf{z}_1 \\
\mathbf{z}_2 \\
\mathbf{z}_3 \\
\mathbf{z}_4
\end{bmatrix}
\]

As mentioned before, this is a representation of the energy transfer between elements. In order to analyse the system’s behaviour it is necessary to obtain the implicit equation matrix.

**MIXED-BOOLEAN IMPLICIT EQUATIONS FROM SWITCHED BOND GRAPH**

The procedure to obtain the state equation is based on the work done by Buisson and Cormerais ((Cormerais and Buisson 2000), (Buisson et al. 2002)). To determine the state equations it is necessary to rearrange the matrix as follows

\[
\begin{bmatrix}
I & 0 & -S_{11}(\lambda) \\
0 & \Lambda_{22} & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\dot{\mathbf{x}}_1 \\
\dot{\mathbf{x}}_2 \\
\dot{\mathbf{x}}_3 \\
\dot{\mathbf{x}}_4
\end{bmatrix} =
\begin{bmatrix}
\mathbf{z}_1 \\
\mathbf{z}_2 \\
\mathbf{z}_3 \\
\mathbf{z}_4
\end{bmatrix}
\]

\[
\begin{bmatrix}
S_{11} & S_{12}(\lambda) & 0 & S_{13}(\lambda) & 0 & S_{14}(\lambda) \\
S_{21}(\lambda) & S_{22} & 0 & S_{23}(\lambda) & 0 & S_{24}(\lambda) \\
S_{31}(\lambda) & 0 & S_{32}(\lambda) & S_{33} & 0 & S_{34}(\lambda) \\
S_{41}(\lambda) & -S_{42}(\lambda) & 0 & S_{43}(\lambda) & S_{44}
\end{bmatrix}
\begin{bmatrix}
\mathbf{z}_1 \\
\mathbf{z}_2 \\
\mathbf{z}_3 \\
\mathbf{z}_4
\end{bmatrix}
\]

This is done to have a set of equations based on the values of the storage elements.

From (5) the implicit state equation can be deduced by using the following constitutive relations for the resistive and storage elements.

\[
D_{out} = LD_{in} \text{ and } \begin{bmatrix}
\mathbf{z}_1 \\
\mathbf{z}_2 \\
\mathbf{z}_3 \\
\mathbf{z}_4
\end{bmatrix} =
\begin{bmatrix}
F_1 & 0 & 0 & \mathbf{F}_1 \\
0 & 0 & 0 & \mathbf{F}_2
\end{bmatrix}
\begin{bmatrix}
\mathbf{x}_1 \\
\mathbf{x}_2 \\
\mathbf{x}_3 \\
\mathbf{x}_4
\end{bmatrix}
\]

After substitutions, the resulting implicit state equation is:

\[
\begin{bmatrix}
I & 0 & -S_{11}(\lambda) \\
0 & \Lambda_{22} & 0 \\
0 & 0 & 0 \\
0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
\dot{\mathbf{x}}_1 \\
\dot{\mathbf{x}}_2 \\
\dot{\mathbf{x}}_3 \\
\dot{\mathbf{x}}_4
\end{bmatrix} =
\begin{bmatrix}
\mathbf{z}_1 \\
\mathbf{z}_2 \\
\mathbf{z}_3 \\
\mathbf{z}_4
\end{bmatrix}
\]

\[
\begin{bmatrix}
S_{11} - S_{11}(\lambda)w_{S_{11}}(\lambda)F_1 \\
S_{12}(\lambda) - S_{12}(\lambda)w_{S_{12}}(\lambda)F_1 \\
S_{13}(\lambda) - S_{13}(\lambda)w_{S_{13}}(\lambda)F_1 \\
S_{14}(\lambda)
\end{bmatrix}
\]

\[
\begin{bmatrix}
S_{21}(\lambda) + S_{21}(\lambda)w_{S_{21}}(\lambda) \\
S_{22} - S_{22}(\lambda)w_{S_{22}}(\lambda)F_2 \\
S_{23}(\lambda) + S_{23}(\lambda)w_{S_{23}}(\lambda)F_2 \\
S_{24}(\lambda) \\
S_{31}(\lambda) + S_{31}(\lambda)w_{S_{31}}(\lambda) \\
S_{32} - S_{32}(\lambda)w_{S_{32}}(\lambda)F_2 \\
S_{33} + S_{33}(\lambda)w_{S_{33}}(\lambda)F_2 \\
S_{34}(\lambda) \\
S_{41}(\lambda) + S_{41}(\lambda)w_{S_{41}}(\lambda) \\
S_{42} - S_{42}(\lambda)w_{S_{42}}(\lambda)F_2 \\
S_{43}(\lambda) + S_{43}(\lambda)w_{S_{43}}(\lambda)F_2 \\
S_{44}(\lambda)
\end{bmatrix}
\]

Where \( H = (L(\Lambda_{34} - S_{44}(\lambda)\Lambda_{44})^{-1})^{-1} \).

This implicit equation is expressed in the form:

\[
E(\lambda)\dot{\mathbf{x}} = A(\lambda)\mathbf{x} + B(\lambda)\mathbf{u}.
\]

This matrix allows the analysis of the system including the properties that usually are not taken into consideration on a classical mathematical approach.

**VALID OPERATIONAL MODES**

Hybrid systems have 2 different operational modes for every commutating element. The number of available
operational modes is $2^n$, where $n$ is the number of switch elements. This is due to the fact that not all the commuting elements change their mode at the same time. In previous works, based on descriptor systems ((Cormerais and Buisson 2000), (Luenberger 1977), (Luenberger 1978)), it could be seen that not all of the available operational modes in a system can be valid. This could be due to the switch element creating interaction between a source and another source, the electrical ground, a mechanical anchorage for example.

In order to have a differentiation between the general representation and the individual operational modes, the following notations are introduced:

- $E(\lambda)_{n_{sc} \times n_{dc}}$ is a square matrix with size $n = n_{sc} + 2n_{dc}$, where $n_{sc}$ is the number of storage elements with static causality, and $n_{dc}$ is the number of storage elements with dynamic causality.

- $E(\lambda)_{n_{sc} \times n_{dc}}$ is the matrix $E(\lambda)_{n_{sc}}$ after the substitution of $\lambda$ values.

From equation (4) it can be seen that some rows corresponding to $\tilde{x}_i$ and $\tilde{z}_d$ could be removed depending on the values of $\Lambda_{22}$ and $\Lambda_{22}$. It must be noted that those corresponding rows of the submatrices $S_{12}(\lambda)$, $S_{24}(\lambda)$ and $S_{25}(\lambda)$ are removed to avoid redundancies when the $\lambda$ element have a zero value. This also applies to the corresponding rows of the submatrices $S_{13}(\lambda)$ and $S_{24}(\lambda)$. Therefore the system’s order could be reduced.

It is simple to find the non-valid modes in systems with a small amount of commuting elements, but it is not for systems with a high number of these elements. Therefore, as the number of commuting elements increases the number of operational mode increases exponentially. Non-valid operational modes can be determined if any of the following conditions can be found on the system:

- If $\text{rank}(sE(\lambda)_{n_{sc} \times n_{dc}} - A \cdot B) \neq n$ when substituting $\lambda$ parameters in equation (6). This is due to the fact that $B$ matrix always have an extra row when there is an interaction between the source elements. Matrix $E(\lambda)_{n_{sc} \times n_{dc}}$ is used due to the commuting elements.

- If a causal loop with value of $+1$ is found, that specific operational mode is non-valid. Controlled junctions allows to break the causal loop and could isolated it to a single operational mode. This depends on the number of controlled junctions interacting with the loop elements. These non-valid operational modes were not considered on previous analysis due to the fact that they were analysed using static causality.

It is simple to identify the non-valid operational modes on systems with a large number of commuting elements. This is done by eliminating all of the configurations that contains the combination of $\lambda$s that creates the first non-valid operational mode.

**EXAMPLE**

As an example, the Graetz bridge is analysed. The model in Fig. 4 is the representation:

![Fig. 4: Graetz bridge](image)

The reference mode on the bond graph model is shown on Fig. 5.

![Fig. 5: Reference mode](image)

In this mode the diodes $\bar{\lambda}_1$ and $\bar{\lambda}_4$ are on ON position while the rest are on OFF position.

From the model the non-valid operational modes can be seen. One occurs when all the commuting elements are set to OFF. Other of the non-valid operational modes occurs when there are two or more commuting element on ON position intersecting on the same 0 junction. And finally, when there are more than 2 commuting elements on ON position on the system at the same time. The reason is that all these modes creates a causal conflict on the system. As a result the following equations are obtained:

$$
\dot{p} = \phi(e - e_c) + a V_1 + b(V_1 + V_2) + c(V_1 + V_3) + d(V_1 + V_4) + q V_5 + e V_2 + f(V_2 + V_6) + g(V_1 + V_5) + h(V_2 + V_6)
$$

$$
f_L = 0 \quad f_R = \phi f_L
$$

where

$$
a = \lambda_1 \lambda_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_6 \quad b = \lambda_1 \lambda_2 \bar{\lambda}_3 \lambda_4 \bar{\lambda}_6
$$

$$
c = \lambda_1 \lambda_4 \lambda_2 \lambda_3 \bar{\lambda}_6 \quad d = \bar{\lambda}_1 \lambda_2 \bar{\lambda}_3 \lambda_4 \bar{\lambda}_6$$
\[ e = \bar{\lambda}_1 \bar{\lambda}_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_5 \bar{\lambda}_6 \quad f = \bar{\lambda}_1 \bar{\lambda}_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_5 \bar{\lambda}_6 \]
\[ g = \bar{\lambda}_1 \bar{\lambda}_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_5 \bar{\lambda}_6 \quad h = \bar{\lambda}_1 \bar{\lambda}_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_5 \bar{\lambda}_6 \]
\[ q = \bar{\lambda}_1 \bar{\lambda}_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_5 \bar{\lambda}_6 \]
\[ \phi = a \oplus b \oplus c \oplus d \oplus e \oplus f \oplus g \oplus h \oplus q \]

\( \oplus \) denotes an xor gate behaviour.

From there, the junction structure matrix is obtained

\[
\begin{bmatrix}
\phi & 0 & 0 \\
0 & \phi & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
p \\
f_L \\
V_1
\end{bmatrix}
= S
\begin{bmatrix}
f_L \\
p \\
e_R \\
E \\
V_1 \\
V_2 \\
V_3
\end{bmatrix}
\]

where

\[
S = \begin{bmatrix}
0 & 0 & -\phi & a+b+c+d+g & b+e+d+f+h & c+e+g+h+q & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
\]

\[
\varphi = \bar{\lambda}_1 \bar{\lambda}_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_5 \bar{\lambda}_6 + \bar{\lambda}_1 \bar{\lambda}_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_5 \bar{\lambda}_6 + \bar{\lambda}_1 \bar{\lambda}_2 \bar{\lambda}_3 \bar{\lambda}_4 \bar{\lambda}_5 \bar{\lambda}_6
\]

Resulting from this the implicit equation

\[
\begin{bmatrix}
\phi & 0 \\
0 & 0
\end{bmatrix}
\begin{bmatrix}
p \\
f_L
\end{bmatrix}
= \begin{bmatrix}
\frac{\varphi R}{L} & 0 \\
0 & -\frac{\varphi R}{L}
\end{bmatrix}
\begin{bmatrix}
p \\
e_R
\end{bmatrix}
+ \begin{bmatrix}
\phi & a+b+c+d+g & b+e+d+f+h & c+e+g+h+q \\
0 & 0 & 0 & 0
\end{bmatrix}
\begin{bmatrix}
E \\
V_1 \\
V_2 \\
V_3
\end{bmatrix}
\]

If any of the non-valid combinations of the commuting elements is introduced on this matrix, there would be a line full of zeros except for the values corresponding to the sources.

**CONCLUSIONS**

In this paper the controlled junctions are selected to represent the commuting elements. This representation shows the effects when there is a change of its configuration, introducing a dynamic causality. A new notation to represent hybrid bond graph systems is introduced and allows to represent all the different behaviours in one matrix.

This representation allows a simple identification of the non-valid operational modes. It is important to identify the non-valid operational modes on systems with a great number of commuting elements to avoid spending time on the analysis of these modes.

The example given shows that the non-valid modes can be detected from the model, nevertheless it can also be detected on the junction structure matrix in the case where the model is not available.

**REFERENCES**


HYBRID SYNCHRONIZATION IN THE DCB BASED ON UNCOORDINATED CHECKPOINTS

Flávio Marques Migowski Carvalho  
Agência Nacional da Aviação Civil  
Distrito Federal, Brasília, Brasil  
email: fmigowski@gmail.com

Braulio Adriano de Mello  
Universidade Federal da Fronteira Sul  
Santa Catarina, Chapecó, Brasil  
email: braulio@ufsc.edu.br

KEYWORDS  
Uncoordinated Checkpoint, Hybrid Synchronization, DCB, Rollback.

ABSTRACT

Hybrid timing synchronization models that combine synchronous and asynchronous components are error prone while executing rollback operations. Mechanisms such as checkpoints are able to guarantee the correctness of these models when the rollback operations of asynchronous components are unavoidable. Checkpoints are saved states of the components in order to recover a previous and consistent state in case of violations of the time constraints. This work presents the specifications and the implementation of uncoordinated checkpoints for supporting rollback operations of asynchronous components based on the DCB architecture. Uncoordinated checkpoints allow us to avoid low performance of the simulations by reducing the overhead of control messages.

INTRODUCTION

In distributed simulations, the different parts of the model execute distinct sequences of events and they do not have their local times synchronized. As each event of each component is able to schedule new events which send new messages to other components, then the synchronization mechanisms are needed to deal with violations of Local Causality Constraints (LCC). One of the issues is the management of checkpoints to execute rollback in case of LCC violations into the asynchronous parts. The recovery from timestamp order violations could be done by a checkpoint-based rollback.

There are studies to improve the management of checkpoints, for example, by reducing the checkpoint time overheads (Soliman and Elmahraby 2002), and selecting the best positions for checkpoints instead of taking them on a periodic basis (Quaglia 2001), and recovering (Cortellessa and Quaglia 2001) by using non-blocking checkpoints (Sato et al. 2012) to improve the efficiency or working with incremental checkpoints to reduce the overhead (Feng and Lee 2006). The management of checkpoints is more complex when the system combines synchronous and asynchronous components (hybrid systems). Solutions such as the mapping of the conservative HLA synchronization interface onto the optimistic one (Santoro and Quaglia 2012) work on the interface of the components. Due to the complexity of real-world systems, more recent studies have proposed solutions to deal with distributed and heterogeneous systems (Reynolds 1988) based on checkpoints and rollbacks for failure detection and failure recovery (Malensek et al. 2013) in distributed simulations.

Besides these issues, heterogeneous models frequently combine synchronous and asynchronous components. As synchronous components cannot rollback their local time, then the simulator must integrate functionalities to keep the model consistent even in the case of rollback of the asynchronous parts. This goal is more complex when the model has components with diversity of modeling languages and their functional interfaces do not match directly to each other.

This work presents the integration of functionalities into the DCB (Distributed Co-simulation Backbone) (Mello et al. 2005) to manage rollback operations based on an uncoordinated checkpoints approach. According to the features of the DCB architecture to support hybrid models, the integration of skills to manage uncoordinated checkpoints is mandatory to guarantee the independence of the heterogeneous components. Besides the susceptibility to the domino effect and to the orphan messages, the checkpoint management strategies integrated into the DCB are able to minimize these effects by reducing the number of control messages drastically.

In the remainder of the paper the Background section presents the main issues of DCB and rollback strategies. The specification of the functionalities, implementation of uncoordinated checkpoints into the DCB and results are presented in the next sections. Following are the conclusions and then references are presented.

BACKGROUND

The DCB (Distributed Co-simulation Backbone) (Mello et al. 2005) is a simulation architecture for heterogeneous
discrete simulation model. It offers resources for cooperation of distributed and heterogeneous components preserving the original code and internal configuration of the components.

The DCB encapsulates adaptation layers for communication and synchronization among heterogeneous and remote components, so that the code for components becomes completely independent from the DCB communication mechanisms. Four main modules, as shown in Figure 1, execute the management of the cooperation process among the model components. They are: DCBS (DCBSender), DCBR (DCBReceiver), DCBK (DCBKKernel), and Gateway.

![Figure 1: DCB Architecture](image)

There are four main modules in the DCB architecture: DCB Receiver (DCBR), DCB Sender (DCBS), DCB Kernel (DCBK) and the gateway. The main gateway task is the interface translation of components. It provides an interface to the other components and it offers data translation services for supporting incompatible interfaces due to the heterogeneity. The DCBR handles the messages management received from other components. The DCBS manages the sending of messages generated in the local element. DCBS and DCBR work together to govern the Local Virtual Time (LVT) of elements. The DCBK manages the message exchange services and the Global Virtual Time (GVT).

All modules of the DCB execute only local operations except for the DCBK. It manages the local or remote connections with the other components based on the configuration sets. All those modules work in a cooperative way to guarantee the correctness of the simulation. In addition, models with hybrid synchronization are supported by the DCB architecture. Its functionalities allow combining synchronous and asynchronous components. Synchronous components are not allowed to execute rollback operations.

Since the generation of the configuration files is an onerous task, the DCB integrates a module for automatic generation of the components configuration for their integration into heterogeneous and distributed models. This module is split in two main parts: the distributed repository; and the automatic configuration of models. The first part, a distributed repository manager of components, offers location transparency of components and automatic collaboration among distributed repositories to broadcast components search actions. The second part allows the automatic configuration of input and output ports of the components and the interface translation when the sender and receiver components have heterogeneous interfaces. The new functionalities for supporting rollback operations presented in this work are compliant with the current DCB functionalities.

**Rollback based on checkpoints**

Rollback operations must be executed in case of violations of LCC (Local Causality Constraint) by one or more components (Elnozahy et al. 2002) and they can affect only asynchronous components. There are two main methods to execute rollback operations: checkpoint-based and log-based (Koo and Toueg 1986).

The log-based rollback could be supported by optimistic protocols or pessimistic protocols. The optimistic protocols allows for the reducing of the control messages overhead (Valchanov et al. 2008). However, they do not avoid the orphan messages thereby increasing the processing load of the garbage collection operations. The pessimistic protocols allow us to avoid orphan messages, however they increase the overhead. Some studies propose solutions which combine optimistic and pessimistic protocols. These studies make efforts to integrate the positive aspects of both protocols but they do not have a stronger effectiveness (Soliman and Elmaghaby 2002).

Protocols based on a checkpoint have three main categories: uncoordinated checkpointing; coordinated checkpointing; and communication-induced checkpointing. In the uncoordinated protocols approach, each component creates its own checkpoints independently. This feature is important to keep the independence among components according to the DCB architectural features. In fact, besides the complexity to treat the domino effect (Koo and Toueg 1986) and that some of the checkpoints could not be part of a global consistent state, the uncoordinated checkpoint approach is better than other approaches to integrate functionalities in the DCB for supporting rollback operations. The main reason is that the capacity to reduce the overhead is a great advantage for distributed models.

Coordinate checkpoints (Moreira et al. 2005) require the processes to organize the checkpoints of all components to compose a consistent global state. They are not susceptible to the domino effect. However, they increase the overhead due to the control messages. The overhead is the main disadvantage. Although the coordination requires more cooperation among all the components of the model to generate secure checkpoints, it makes the rollback operations simple since all processes return only to the most recent checkpoint in case of rollback. The
most recent checkpoint is always enough to get back the consistency of the simulation. The oldest checkpoints can be discarded.

The communication-induced checkpoints allow for avoiding the domino effect without coordination among all the components. In order to coordinate the checkpoints, the components create local or forced checkpoints. The local checkpoints are independent and they are not a part of a consistent global state. A guaranteed global time line is mandatory for creating the forced checkpoints. They use information of the communication messages of components to guarantee the consistent global state.

**MANAGEMENT OF CHECKPOINTS BY THE DCB**

An uncoordinated checkpoints approach was chosen to be integrated into the DCB architecture due to the following DCB features: components are completely independent (they do not communicate directly with each other); DCB supports heterogeneous and distributed models; and the interoperability among different modeling languages. In spite of the uncoordinated checkpoints being susceptible to the domino effect and requiring garbage collection, the cost of the coordinated protocols could be excessively onerous on distributed and heterogeneous models.

The DCB is responsible for guaranteeing the consistency of the checkpoints based on the information from the communication messages and to manage the rollback operations in case of LCC violation. Moreover, asynchronous components must be able to create their own internal checkpoints to recover previous internal states in case of rollback operations. Currently, the DCB does not integrate functionalities to identify safety states in order to create checkpoints only in consistent global states. This goal is seen as future work. These functionalities will be useful to reduce the impact of the domino effect and the impact of the garbage collection operations.

![Figure 2: Example of worthless checkpoint](image)

An arrival of the message $m_k$ with a timestamp less than the LVT of the receiver component, illustrated in Figure 2, works as a trigger for the third component to come back to the first checkpoint before the timestamp of $m_k$. As the checkpoint is before the sending time of the message $m_l$, then $m_l$ is discarded. Due to the discarding of $m_l$, then the first component comes back to the previous checkpoint which is at the beginning of the simulation in this example. These actions also require the discarding of the message $m_l$. This example shows a domino effect scenario where all components are triggered for rollbacking until the simulation time of zero.

In the same scenario illustrated in Figure 2, the second checkpoint of the third component created after the local time $x$ is worthless and it can be discarded by the DCB without affecting the simulation.

Figure 3 illustrates an inconsistent state where an asynchronous component sends the message $m_i$ to a synchronous component whose LVT is greater than the timestamp of $m_i$. Models with hybrid synchronization do not accept rollback operations for synchronous components. When a synchronous component receives a message with a timestamp less than its current time, the simulation is considered to be failed. Due to that, the DCB takes into account only the LVT of the asynchronous components plus the lookahead to update the GVT (Global Virtual Time) (Fu et al. 2013).

![Figure 3: Example of inconsistent state](image)

As the DCB takes into account only the LVT of the asynchronous components for updating the GVT, then the rollback operations will never cause reducing of the GVT. Due to that, the synchronous components always keep their LVT consistent in relation to the GVT. When a synchronous component receives a message with a timestamp less than the LVT then the DCB sees a deadlock and the simulation is canceled. The internal functionalities of the DCB which manage the rollback operations also works to prevent the deadlocks.

Uncoordinated checkpoints allow each component to build checkpoints at distinct LVT. There are two main advantages: reducing of the overhead caused by control messages to coordinate the checkpoints; and the support of the DCB features to keep the independence of the components. The coordinated checkpoints require specific functionalities to manage the time advance of the LVT of each component in a coordinated way. These functionalities are not in compliance with the above
advantages mainly for distributed and heterogeneous models.

**DCB functionalities for rollback supporting**

The rollback procedures are encapsulated by the DCBS and DCBR modules. They are responsible for managing the received and sent messages by the components. Each component has its own copy of these modules and its own gateway. These modules and the gateway are responsible for communicating processes among the components based on the input and output ports configuration. Whenever the gateway of a component identify an anti-message, then the timestamp of the anti-message is used to verify whether there are messages in the local buffer which must be discarded according to the rollback process. This operation is executed by the DCBR.

In Figure 4 there is presented the simplified code of the gateway functions to manage the rollback operations. Whenever there are messages in the local buffer with the timestamp less than the timestamp of a recent received anti-message, the DCBS sends new anti-messages to the components which also need to execute the rollback operation. The global rollback finishes when the last consistent checkpoint is reached and there are no more anti-messages to be sent by the components.

Line 4 of Figure 4 shows whether the component is processing the rollback operations or not. If true, the LVT advance is prohibited while the component does not finish the rollback execution and restarts the simulation advance.

```
1. A0 = DCBR.getAttributeReceived("444.3");
2. if (A0 != null) {
3.    DCBR.getAttributeRemove(A0);
4.    Fed.rollback = true;
5.    checkpoint = Fed.getCheckpoint(A0.LVT);
6.    if (checkpoint != null) {
7.        DCBS.sendMessage(checkpoint);
8.        Fed.rollback(checkpoint);
9.        Fed.setChatLVT(updateLVT(A0.LVT));
10.       Fed.setRecvText(A0.Value);
11.       Fed.rollback = false;
12.    }
13. }
```

**Figure 4: Rollback functions of DCB**

The simplified class diagram in Figure 5 shows the main methods which implement the new functionalities for supporting the uncoordinated checkpoints into the current version of the DCB. The ExchangeMsg class belongs to the case study presented in this work.

The DCBS keeps the messages sent by the component in a local buffer in order to manage receiving anti-messages in case of rollback operations. The DCBR logs the history of sent and received messages for supporting the gateway to communicate with the local component.

In Figure 5 the component implemented by the class ExchangeMsg is able to create new checkpoints using the method setCheckpoint() and to execute the rollback operations based on previous checkpoints. As the DCB is using uncoordinated checkpoints, the components have no time constraints to define the states on the time line to create new checkpoints.

**Figure 5: DCB structure for supporting uncoordinated checkpoints**

Asynchronous components are also required to be able to manage internal rollback operations. As the DCB communicates with the components only by the communication ports of the interface, then the internal processing of the components is not affected by the functionalities of the DCB. The DCB manages the exchange of messages and anti-messages but it does not manage the internal processing of the components. If a component does not create internal checkpoints with information about its internal processing, it must be integrated into a model as a synchronous component. In general, asynchronous components implement internal procedures to process anti-messages in case of rollback.

**IMPLEMENTATION AND RESULTS**

We integrated the proposed uncoordinated checkpoints solution into the current version of the DCB and we implemented a simple heterogeneous model for validating purposes. The model represents a message exchange system (ExchangeMsg) and it integrates synchronous and asynchronous components. The following components were created for the case study:
• C5 and C9 (synchronous)
• C6, C7 and C8 (asynchronous)

All the components have the same internal behavior, they are independent and they communicate among themselves for exchanging simple text messages. The simulation time $\theta$ is seen by the DCB as the first checkpoint and it is the lowest limit to the rollback operations. The DCB generates a new checkpoint for each component in period intervals of ten (10) received or/sent messages. The current LVT less than the LVT of the last checkpoint must be equal to or less than 5000 time units. We have not applied strategies to calculate the best intervals between the checkpoints because the main goal of this case study is to validate the rollback operations. So, it was enough to generate states with a time constraints violation for asynchronous components for validating purposes. The estimation of the best intervals between checkpoints is seen as future work.

During the execution of the simulation the DCB identified messages received by the asynchronous components whose LVT were greater than the timestamp of the messages (LCC violation). One of these situations is illustrated in Figure 6 and it has the following sequence of steps:

- LVT(C6) = 1000 (LVT of Component 6 is equal to 1000) and C6 sends the $m_6(2000)$ (message $m_6$ with timestamp equal to 2000) to C7;
- When LVT(C7) = 2000 then C7 receives $m_6$ from C6;
- LVT(C8) = 400 when C8 sends $m_8(500)$ to C6;
- DCB detect LCC violation and it sends $\alpha(500)$ (anti-message with timestamp 500) to C6;
- Then the Gateway of C6 starts the rollback process.

![Figure 6: Scenario of imminent rollback](image)

After sending the anti-messages and having started the rollback process, the following steps are executed:

- The LVT advancing of C6 is interrupted;
- DCBS of the C6 sends $\alpha(1999)$ to C7.

Then, C7 discards the message $m_6$ sent by C6 and returns to the nearest checkpoint before the current LVT. After the rollback processing, C6 restarts the time advance process and it receives the message $m_1(500)$ sent by C8.

Whenever an asynchronous component with $LVT(t)$ sends a message to a synchronous component then the $LVT(t+i)$ becomes the new lowest limit for rollback operations. This restriction is required to guarantee the consistency of the synchronous components. This functionality is integrated into the DCB. In this scenario, when an asynchronous component needs to return to a state before $LVT(t+i)$ then the simulation is considered to be failed and the model must be reviewed. These issues are foreseen in the new case studies.

CONCLUSIONS

This work presented a set of functionalities for supporting rollback operations by the DCB of hybrid timing synchronization models based on an uncoordinated checkpoints approach. These functionalities enable the DCB to create uncoordinated checkpoints and to manage rollback operations keeping the time correctness of the components.

We chose an uncoordinated checkpoints approach in compliance with the main DCB features about the independence of the components, the supporting for heterogeneous and distributed models, and the interoperability among different modeling languages.

In our proposed solution the components are free to define their own checkpoints without concerning the timing coordination with the other components. This feature allows for the reducing of the management messages and guarantees the correctness of the simulation. The future works include the specification of strategies to reduce the creation of worthless checkpoints, and the definition of strategies to find the best intervals between two checkpoints based on induced-techniques and information collected from the simulation messages, and performance analysis between scenarios based on uncoordinated and coordinated checkpoints.

Acknowledgements

Braulio Adriano de Mello acknowledges the support from the Brazilian research funding agency CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), proc n. 1835-14-9.

REFERENCES

Fu D.; Becker M.; and Szreberieka H., 2013. “On the Potential of Semi-conservative Look-ahead Estimation in Approximative Distributed Discrete Event Simulation”. In


META-RESOURCE MODELS
A META-MODEL FOR CONSISTENT & AUTOMATIC SIMULATION MODEL SELECTION

Sangeeth Saagar Ponnusamy  
Airbus Operations SAS & CNRS, LAAS  
316, Route de Bayonne  
Toulouse-31060  
sangeeth-saagar.ponnusamy@airbus.com

Vincent Albert  
CNRS, LAAS  
7 Avenue de Colonel Roche  
Toulouse-31400  
valbert@laas.fr

Patrice Thebault  
Airbus Operations SAS  
316, Route de Bayonne  
Toulouse-31060  
patrice.thebault@airbus.com

KEYWORDS  
Modeling, Simulation, Ontology, SysML, Reasoning.

ABSTRACT

A meta-model of modeling abstractions is presented and discussed in the context of using simulation as a means for system verification and validation activities. Extending the classical results of presenting modeling as a relevance reasoning problem, abstractions are classified to build an ontology based on the concepts of teleological modeling and relevance reasoning. The domain model is built in the standard ontology tool of Protégé to exploit the reasoning and inference capabilities to build a model abstraction library. Lattice structures of model instances are formed and a recursive algorithm is implemented as an activity diagram in SysML for automated and consistent model selection. The approach is presented with a battery model example from the literature. Challenges and future work in implementation of such a semi-formal approach in model selection is briefly presented in the context of improving fidelity of simulation in the industry.

1. INTRODUCTION

In using Modeling and Simulation (M&S) as a means for the system Verification & Validation (V&V), often the difficulty is finding and implementing abstractions of the system being simulated with respect to the simulation requirements. Such difficulties gives raise to the problem of simulation fidelity i.e. the effectiveness of simulation in reproducing the reality for the System Under Test (SUT) validation. In the field of artificial intelligence, this model construction problem is posed as a reasoning problem (Iwasaki and Levy 1994) i.e. inclusion of relevant information about the system being modeled and reasoning about them. However, identification, organization and further classification of these information i.e. abstractions for models simulating a phenomenon are often a tedious task. This is even more so true in modeling complex systems and it becomes imperative that this identification and organization of domain knowledge about the model be (semi)automated in a Model Based System Engineering (MBSE) context.

Ontology, which is a formal representation of a set of concepts within a domain and the relationships between those concepts, could serve as an answer to this problem of building a domain model due to its inherent reasoning and knowledge exploitation capabilities. This domain model, also called meta-model, serves as a common framework for better understanding and making domain assumptions explicit. Since a model can be interpreted as a set of concepts with some relationship between them, ontologies could be used in their representation, organization and exploitation by the practicing engineers with ease in this MBSE framework.

In this paper, a domain model of modeling abstraction is proposed and implemented in Protégé tool, (Protégé). This domain model serves as a standard template for instantiation by system designers and simulation users. The reasoning and querying abilities are used by the model developer over these instances to identify and classify abstraction to build a model abstraction library. The recursive algorithm (Lickly 1992), implemented in a SysML activity diagram is then used to identify a consistent yet simple modeling abstraction to be implemented by the model developers. A simple example of a battery model from the literature is taken as a case study.

2. ONTOLOGIES & REASONING FOR M&S

Reasoning in general is the process of deriving facts that are not explicitly stated. In (Iwasaki and Levy 1994) modeling abstractions were discussed as inclusion of relevant information about the system being modeled and reasoning about them. Based on this relevance reasoning, Levy et al (Levy et al 1997) proposed a recursive procedure to find the consistent yet simplest model from an existing model library. This library is assumed to be well formed i.e. modeling assumptions behind each model is clearly identified and classified. Unfortunately this is not often the case especially when system development and model development are done by different entities who do not necessarily share the same domain knowledge and its associated vocabulary. This problem is further compounded in system V&V activities by simulation when the models of system need to be developed according to some user defined scenarios. Thus the problem of fidelity too can be posed as a reasoning problem i.e. inclusion of the relevant information according to a given scenario and reasoning about them.

In order to ensure this view point consistency, standardization of knowledge exchange and its exploitation, ontologies are proposed in this paper for developing models with required fidelity. Originally intended for semantic web, ontology practices has been increasingly used to improve semantic interoperability and consistent modeling in a MBSE framework (Jenkins 2012) (Man et al 2009). Greves et al (Greves 2009) discussed a reasoning aided MBSE approach by integrating ontology and SysML, a general purpose system engineering language. Ontology, as a means of incremental knowledge addition will help in formalization of these inclusion relations by domain experts over time with their validation experiences. The availability of the standard Web Ontology Language called OWL (OWL) and tools such as Protégé with its query and reasoning capabilities makes it...
an attractive option to perform these activities. In Protégé complex concepts can be incrementally built up from simpler concepts using rich set of operators, after which plug-in reasoners such as Fact++, Hermit (Hermit) are used to draw inferences and check consistency. Reasoners infer this relationship by reification, a concept in logic where an instance of a relation is made the subject of another relation. The inferred ontology can be queried for specific needs with SPARQL, a query language which is used to retrieve and manipulate data stored as Resource Description Framework (RDF), a standard for the semantic web. Queries are constructed in triple pattern of subject, predicate and object with conjunctions, disjunctions and optional patterns such as to filter, sort etc.

Broadly the contribution of this paper is twofold, building a model abstraction library based on ontology reasoning and exploitation based on SysML implementation of model selection and automated assembly based on ontology queries. The paper is structured as follows, the classification of abstractions are presented.

3. A META-MODEL OF MODEL ABSTRACTIONS

A meta-model for M&S must include different viewpoints of the system being modeled in a teleological perspective. Since a model is an abstraction i.e. simplification of a system and intends to represent a system phenomenon at some operating condition through some quantities and relationship between them, the meta-model need to have different classes of abstraction to represent these aspects. Thus the objective of such meta-model is to translate system knowledge usually expressed in natural language to explicit and model based form for standardized exchange between the stakeholders.

3.1 SBFIO Ontology

In understanding and design of complex systems, teleological modeling in the form of Structure, Behavior and Function (SBF) framework is important. The SBF ontology was proposed as a set of distinct activities in design science as a basis for modeling (Garo 2001). This can be extended with notation of interface (I) and Operating mode (O) to describe interconnected system with different modes of operation. The SBFIO ontology is briefly explained below.

System is composed of a Structure (S) i.e. architecture in the form of system-subsystem-equipment-component hierarchy i.e. a System is composed of subsystem which in turn composed of equipment and equipment is composed of component, e.g.: battery is composed of terminals, resistor, capacitor etc.

System exhibits a Behavior (B), e.g.: battery exhibits voltage discharge as function of time.

System performs a Function (F), e.g.: battery provides power.

System communicates via an Interface (I) which are ports of exchange, energy and data, between physical and cyber systems respectively, e.g.: battery connected to electrical circuit via terminal interface.

System operates in an Operating Mode (O), Mode is a partition of state space of system () and for interconnected system, Operating Mode refers to causal dependency between modes of interconnected systems or components. e.g.: when battery is in mode ‘charging’ the electrical circuit is in mode ‘off’, when the battery is in mode ‘discharging’ the electrical circuit is in mode ‘on’.

In general, a model is essentially a representation of any or all of these system perspectives. In addition, behavior or consequence is essentially an outcome of relationship between model quantities which in turn characterizes a function. Their relationship is briefly illustrated in figure 1 with other relationships and concepts of the domain model being hidden for the sake of brevity.

Figure 1: SBFIO Framework

For example, consider calculate altitude function performed by the aircraft navigation system at cruise which is characterized by the quantity, altitude (ft) but from two different sources namely, radio altitude and GPS height data for redundancy reasons. This domain model will help in maintaining design consistency such as: all altitude quantities has same syntaxes e.g.: unit as ft and semantics e.g.: absolute height not relative height. These ontological relationships help in top down traceability from high level functions to low level behavior with corresponding architectural granularity. These concepts and relationships are being implemented in the context of a system V&V by simulation ontology where the entire processes, from fidelity requirements capture to its implementation via a vis consistent modeling abstractions are covered.

3.2 Classification of Abstractions

A model is built to represent one or more system viewpoint described in section 3.1 via abstractions. There exist different taxonomies for abstractions employed in M&S by (Frantz 1994). In our approach, modeling abstractions are broadly classified into four classes namely, architecture, data, computation and time (Albert 2009). For the sake of brevity, the class definitions are not discussed in detail since the focus is, for a given a class description, how to reason and select corresponding modeling abstractions. A part of this class definition is implemented in Protégé and intuitively one can see that architectural class corresponds to structural viewpoint whereas computation class corresponds to behavourial viewpoint. Only computation, architecture and data dimension will be discussed henceforth.

In general, the abstraction classes are identified $c \in \mathbb{R}$, where $\mathbb{R}$ is the domain model shown in figure 1. Consider a model $M$, defined by an abstraction operation $\alpha$, where $\alpha'$ is a member of the abstraction class $\alpha'$ set as described above. This model definition is valid for a certain condition called Operating Condition (OC). For example, an aerodynamic
model with abstraction of only laminar flow is valid for a range of Reynolds number, $Re<Re_{\text{limit}}$. The hierarchy of abstractions is related by binary relation forming a partial order ($\preceq$) as follows.

$$M_0 \preceq_1 M_1 \preceq_1 \ldots \preceq_1 M_N$$  \hspace{1cm} (1)

where $M_0$ refers to concrete model and $n=1..N$ are possible abstractions.

The model abstraction library lists the models and their corresponding abstraction and operating conditions as described in the Table 1 below. The abstractions defined manually by the developer or user are indicated by the ‘*’ sign and those which are inferred then by reasoning capabilities of the ontology to complete this table to the extent possible, are denoted by ‘+’ sign.

<table>
<thead>
<tr>
<th>Model</th>
<th>Abstraction</th>
<th>Operating Condition</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M_0$</td>
<td>$a^0_1 \preceq_1 a^0_2 \preceq_1 \ldots \preceq_1 a^0_N$</td>
<td>OC$_1$</td>
</tr>
<tr>
<td>$M_1$</td>
<td>+ $\preceq_1$</td>
<td>OC$_2$</td>
</tr>
<tr>
<td>....</td>
<td>+ $\preceq_1$</td>
<td>OC$_2$</td>
</tr>
<tr>
<td>$M_N$</td>
<td>+ $\preceq_1$</td>
<td>OC$_2$</td>
</tr>
</tbody>
</table>

* defined, + inferred

The models described by such a partial relation forms a lattice. Lattice or Hasse diagram is a mathematical diagram of this partial order relation. Such models described over lattices are grouped based on the abstractions. Since a valid abstraction is an operation from a concrete model to an abstract model, where, whatever is true about the concrete model is true in the abstract model but the reverse is not necessarily true, the properties can be inferred from such inheritance relations. From Eq. (1) for models $M_n$ and $M_{n+1}$ and their requirements $\varphi$ defined over some temporal logic such as Linear Temporal Logic (LTL) or Signal temporal Logic (STL), if

$$M_{n+1} = \{\varphi_{p=1,P} \} \Rightarrow M_n = \{\varphi_{p=1,P} \}$$  \hspace{1cm} (2)

Thus for an abstraction belonging to the same class $a^0_{i=1..n}$ arranged over the lattice, implementation of an abstraction $a^0_{i+1}$ also mean the implementation of abstraction $a^0_i$ due to partial order relation $a^0_i \preceq a^0_{i+1}$. The model abstraction library is thus filled based on these inheritances and dependencies identified by reasoning over the partial order relations. These inclusion relations are exploited to fill the modeling abstraction library and this approach is illustrated with a battery example in next section.

4. APPLICATION CASE

The application case is a battery system similar to the one described in (Levy et al 1997). The battery is connected to a solar panel of a satellite and the function of the battery is to provide power to the panels when the satellite is at the far end of earth without the sunlight.

It is known that a phenomena exhibited by the system can be modeled in different ways. Thus the battery can be modeled in different perspectives (e.g.: model voltage phenomena, charge level or a combination of both) and for each perspective it can be modeled in varying granularity of details (e.g.: voltage is independent or dependent of charge level). Every such model may correspond to different operating condition and the challenge is to find an abstraction consistent with the required operating conditions and phenomena.

The model abstraction library based on table 1 for this application case has models with Voltage (V) as output with different abstractions on ChargeLevel (CL), time (t), Temperature (T). The model ids are given by the following set, $M_{n+1} = \{\text{Constant Voltage, Binary Voltage, Normal Degrating-1, Normal Degrating-2, Charge Sensitive, Temperature Sensitive}\}$. The Operating Condition (OC) corresponds to state of damage and rechargeable conditions. For this case, there are only two conditions namely \{not damaged\} and \{not damaged, rechargeable\} denoted by OC$_1$ and OC$_2$ respectively.

Let us denote a class and its instance by a notation $\text{Class: Instance}$. Consider a sample model $M_5$, $\text{Model:Charge:Voltage}$ which describes the evolution of voltage as function of charge level and time under a condition not damaged. The Quantity: ChargeLevel and Quantity:Time is defined to characterize the battery function, $\text{Function:Charge}$. An instance could be defined or inferred and the objective is to minimally define these instances and infer the rest. For example, a model with $\text{OperatingCondition: rechargeable}$ upon inference becomes not damaged too. This rule is encoded in ontology through a subsumes relation such that if OC$_1$ subsumes OC$_2$ then OC$_1$ = OC$_2$. Similarly other domain specific rules could be implemented by domain experts and such template will be useful for other stakeholders to find the dependencies through inference.

In addition, queries can be made on the instances to extract required data or match related data. For example, models could be grouped under an assumption classes based on the output quantity, Voltage (V) in this case. Then, using SPARQL queries, all models having same outputs can be extracted and grouped. Similarly, instances of a class $\text{Parameter:Dependancy}$ defining the quantities characterizing the function under an operating condition can be queried to answer teleological questions such as listing functions which depends on same parameters etc.

In the following section, only a few abstractions for each class are explained and this method can be extended for others too, provided a hierarchy can be built with binary relationship between them as described by Eq.1.

4.1 Architectural Abstractions

Architecture relations such as system-subsystem-equipment-component are expressed through Structure Composed of relationship. For example the battery system is composed of component such as terminals, switches etc. An instance $\text{Model:Binary_voltage_Model}$ with the relation Structure Composed of to another instance $\text{Structure part: Binary_voltage_Model Terminal}$ which in turn related to other instance such as port etc. Intuitively, a simulation user requirement of simulating a battery port implies simulation of its parent system.

4.2 Data Abstractions

Similar to previous example, a hierarchy of data types could be created using data part property. A simulation model data type abstraction is deemed valid if the data type is at least less abstract than required by the user. For example, describe data types (DT) as Float $\preceq$ Int $\preceq$ Boolean, and the simulation user required data type DT$_{user}$ as Int and
that of model developer, $D_{dev}$ as float. It is inferred that $Int$ is also a float and hence the data type abstraction is deemed valid. These lattice declarations could be extended to other concepts in the context of static model analysis for mitigating model composition errors (Lickly et al 2011).

4.3 Computation Abstractions

Consider a type of computation abstraction such as accuracy which is the difference between exact solution and approximate solution due to modeling abstractions of the behavior. One such abstraction is the Model order which refers to the degree of freedom or in other words the ability of model to capture the rate of change of the dynamics. The model dynamics defined with same input quantities could be related with Model Order part relationship with the dimension of its space i.e. the complexity of the model. Let the order be defined as, $\Theta : M \rightarrow N$, where $N$ is set of natural numbers. If $(M_2)\subseteq(M_1)$, $M_2$ is more capable than $M_1$ and it intuitively implies the former model captures the dynamics of the later as well. Hence the model abstraction at higher order infers the model simulates lower order dynamics too.

Consider, battery example which models the output voltage as function of different parameters based on their Input-Output (IO) relations. The abstraction hierarchy $q_{10}^m$ corresponds to the number of inputs for the function, $f_m$ where $m \in N$ is the order of function. As described in section 3.2.1, for the models of same IO inputs, the hierarchy can be further decomposed on the model order. In the Normal Degrading-1 & 2 case, the second order model, $M_4$ also simulates first order behavior given by the model $M_3$.

Similar such reifications i.e. information enrichment can be done for members of other classes such as architecture etc. as described in sections 3.2. Upon completion of the model abstraction library, the next task is to select the model consistent with requirements which necessitates the construction of the lattice which will be explored by the recursive algorithm. The lattice structure can be generated by a lattice plug-in or Formal Concept Analysis (FCA) tools such as Lattice Miner where the abstraction library is given as input in the form of objects and attributes. Similar to the lattice described in (Levy et al 1997), the generated lattice for models with Voltage as output is shown in the figure 2. The objects i.e. models are noted in red and attributes are noted in blue and the inclusion hierarchy can be seen. For example, the Model: Temperature Sensitive is modeled by temperature, CL and time whereas the Model: ChargeSensitive does not model temperature effect. In other words the latter model is an abstraction of the former or lower the lattice element higher is the complexity.

![Figure 2: Lattice for Voltage Assumption Class](image)

Similar lattice can be generated for other consequence quantities or any other assumption classes.

5. AUTOMATED MODEL SELECTION

In this section, a SysML implementation of the recursive algorithm to identify a necessary and sufficient simulation model is presented. This implementation consists of block diagrams to define the domain model and activity diagrams for the description of the algorithm. The algorithm is executed over the instantiated domain model i.e. model abstraction library previously constructed in section 3.3. The resulting output is a selection of consistent model with necessary and sufficient abstraction which is built in the form of parametric diagram to be directly simulated. The modeling tool used is MagicDraw SysML (NoMagic) with its Cameo Simulation Toolkit plugin for the execution of built models, in our case execution of activity diagram over the instantiated domain model. Also it could be possible to transform ontology models to SysML using OWL2UML plugin in Protégé 4.1 and then initiating the algorithm. It may be noted that the domain model built is same as described in section 3.2, except for the SBF class of functions and the abstraction taxonomy.

The model selection problem is to find a necessary and sufficient consistent model called scenario model i.e. model attribute of ModelSelection class, from the given input of domain theory, i.e. a set of model abstraction from the library, called assumption classes, and a query. A query is characterized as follows:

- a list of quantities, quantity whose value to be predicted by simulating the system,
- a list of exogenous quantities, $E_{input}$ whose elements are assumed to be given and to be outside the scope of the simulation for which scenario model is constructed.

A domain theory is characterized by a set of assumption classes. An assumption class is a set of models which describe the same phenomenon, i.e. having the same output quantity in their consequence based on different and often contradictory modeling conditions. Quantity, as described in section 3.1, is an atomic expression denoting time dependent attributes associated with the participants in a model instance. On the other hand, Consequences are statements that are true whenever the phenomenon represented by the models takes place. Consequences can also be any other logical assertions that are true in a state in which an instance of the model exists.

Activation conditions are statements that indicate when the phenomenon represented by the model takes place by specifying constraints on the participants of the model and on its quantities. The conditions include both structural constraints on the participants as well as constraints on the ranges of quantity values.

Models are related to each other by a refinement/generalization relationship $Rel$. A model can be related to zero or many other models which are simpler i.e. more abstract or complicated i.e. less abstract. It is assumed that every assumption class has a single most complicated model and a single simplest model. In other words the lattice is finite with a minimum and maximum.

These concepts were implemented in SysML and lattice structures are instantiated according to this SysML implementation. The selection algorithm implemented as activity diagram, which is not shown here due to lack of
space, is then used to recursively find the consistent yet simplest model. It may be noted that an activity diagram specifies input to output transformation through controlled sequence of actions and the model selection algorithm is formalized in it and executed. We have used iterative expansion region for list iterations,"readStructuralFeature" and "addStructuralFeature Value" actions for attributes' getters and setters of classes, call behavior actions for modularity and reusability of functions, merge and decision nodes for choices and conditions. The results for the models which correspond to the query Voltage for conditions not damaged is given below in the figure 3. Informally, the algorithm starts with simplest model and progressively adds the assumption according to the requirement until all the necessary assumption classes are added out of which a simplest model is chosen. In this case, the final scenario model is \{battery-damaged, charge-sensitive, accumulation-with-ageing\} and each selection is highlighted in grey at the end of each iteration in the figure 3.

![Figure 3: Model Selection Results](image)

6. FUTURE WORK & CONCLUSION

A possible drawback of our approach is the implementation of class definition in Protégé and model selection in SysML. This is done in order to leverage the flexibility, scalability, query and reasoning powers of ontology with the control flow execution, graphical interface capabilities of SysML. However this approach has limitations in terms of effort and at times redundant. This necessitates an integration of SysML and OWL as remarked by (Greves 2009) and (Wagner et al 2012). Such a mutual transformation between SysML and ontology will help practising engineers to capitalise on their graphical syntax and reasoning capabilities respectively and thereby ensuring seamless design and product V&V activities. In addition, studies are being carried out (Ponnusamy et al 2014, 2015) to extend this approach in the established M&S framework of Experimental Frames proposed by Zeigler (Zeigler 2000).

This domain model approach based on ontologies & SysML could be integrated in the standard M&S process (Thebault et al 2015). Such ontology aided simulation design process will enable different stakeholders in simulation to define, solicit and manage knowledge usable for M&S in a consistent way. Realization of such an objective will help improve the level of confidence in simulation results for the system V&V and help better utilization of simulation resources by selecting the best available resource according to the test objectives.

REFERENCES


WEB REFERENCES

http://hmmt-reasoner.com/
http://www.nomagic.com/
http://owl.man.ac.uk/factplusplus/
http://protege.stanford.edu/
http://www.w3.org/TR/rdf-sparql-query/
A RESOURCE MODEL FOR THE RULE-BASED DYNAMIC BUSINESS PROCESS MODELLING AND SIMULATION

Olegas Vasilecas¹, Diana Kalibatiene¹, Audrius Rima¹, Ilze Birzniece², Peteris Rudzajs²
¹ Vilnius Gediminas Technical University, Saulište 11, LT-10223 Vilnius, Lithuania
E-mail: {diana.kalibatiene, olegas.vasilecas, audrius.rima}@vgtu.lt
² Riga Technical University, Kalku street 1, Riga, LV-1658, Latvia
E-mail: {ilze.birzniece, peteris.rudzajs}@rtu.lv

KEYWORDS
Dynamic business process, resource, business process simulation, resource model, resource perspective.

ABSTRACT
The success of each business depends on its inside running processes. However, traditional approaches and tools used to model, simulate and implement business processes (BP) are static, and therefore, no longer covers the actual needs of the business, which should be dynamic. Therefore business needs tools, which support such dynamic BP modelling and simulation. For the modelling and simulation of dynamic BP we need to know available resources for process simulation. In this paper we analyze resources perspective used for dynamic BP modelling and simulation and propose a resource model for rule-based dynamic BP modelling and simulation. The proposed resource model has been implemented into the resource database and used for the rule-based dynamic BP simulation. The obtained results show the suitability of the proposed resource model and its flexibility to define new attributes of resources.

1. INTRODUCTION

In today’s dynamic business world the success of an enterprise increasingly depends on its ability to react to changes in its environment in a quick and flexible way. A necessity to model and simulate the dynamic business process (BP), which can be changed at BP instance run time, and adopt them to new requirements arises.

Resources make an important part of BP modelling and simulation. The resource perspective of BP refers to the link between the activities defined in the processes and the resources that carry out the work related to them (zur Muehlen 2004; Stroppi et al. 2011). The way in which the work is distributed to the resources is significant to the efficiency and effectiveness of a BP execution. It is necessary to ensure that different activities of a BP are carried out by suitable resources. As stated in (Stroppi et al. 2011; Moore 2002), excessive activity automation and poor design of work assignment strategies are critical issues in workflow projects. Therefore, it is necessary to define resource perspective aspects appropriately.

In this paper we analyze the resource perspective for BP modelling and simulation approaches and propose a resource model for rule-based dynamic BP modelling and simulation. The proposed resource model was implemented into the resource database and used for the rule-based dynamic BP simulation.

The rest of the paper is organized as follows. Section 2 presents related works on dynamic BP and a resource perspective. Section 3 describes the proposed requirements for the resource model and proposed resource model. Section 4 presents an implementation of the proposed resource model. Section 5 concludes the paper.

2. RELATED WORKS ON THE RESOURCE PERSPECTIVE

In this section we present definition of main concepts used in the research and related works important for our proposition.

The analysis of the related works shows that Wikipedia (2015), organizations (Gartner, Inc. (2013), WhatIs.com (2012)) and research papers, as (Adams 2010; Pesic and van der Aalst 2006), define dynamics of a BP as the ability to react to changing conditions (internal and/or external) of operation according to the client’s individual needs in an appropriate and timely manner at process instance run time without negative impact on the process essence or its expected completion.

The analysis of the related works shows that there are several ways to ensure dynamics of BP. Main of them are as follows: context-sensitive (Saidani and Nurcan 2007), event-based (Hermosillo et al. 2010) and rule-based (Pesic and van der Aalst 2006; Mejia Bernal et al. 2010) approaches. Yet, in some cases, it is enough to define context, or use complex event processing, or restrict a BP by a set of business rules or constraints. However, it is not suitable for other cases.

According to (van Der Aalst 2015), simulation provides a flexible approach to analyze BP. Through simulation experiments various “what if” questions can be answered and redesign alternatives can be compared.

According to Dictionary.com (2015), resources are materials, services, personnel, knowledge and other assets used for producing wealth in the organization. Resources can be utilized, e.g. those are resources which are constantly decreasing and have to be supplemented, or employed, e.g. after executing an activity the resource becomes free and available for other activities.

According to the well-known definition of the BP, like presented in (Aguilar-Saven 2003), it consists of activities, which have to be related with concrete
resources to execute or simulate them. Several resources may have the same role, for example we can have three administrators, and several activities may require a particular role (van Der Aalst 2015). The simulation model needs to specify resource requirements and usage. Also the number of resources per role, the selection of resources and the ordering of pending activities need to be specified in the resource model. For example, client needs “wood”, e.g. “oak” “100 m³”. According to these attributes the query for necessary resources can be executed and the necessary resource will be selected.

Another attribute for the resource is its usage cost (Vasilecas et al. 2014). The sum of all resource costs allow to calculate the cost of executing an activity, herewith, to calculate the cost of all BP.

As stated in (van Der Aalst 2015) the duration of activities should be modeled also. In most BP, the average flow time of a case is much longer than the average service time (i.e., the time actually worked on the case). This is due to queuing for unavailable or busy resources. Therefore, activities should have priorities not to delay their simulation because of unavailable or busy resources. Start usage data and end usage data of resources in a particular activity allows us to calculate the actual duration of all process.

It should be able to get information about the current state of resources at any time. It will allow to have a list of free resources and to assign them to activities.

It is evident that BP often has a limited number of resources. For this reason, the competition of processes can arises. Mostly, competition is observed when two or more BP instances start at the same time and use the same resources (Vasilecas et al. 2014). Therefore, the simulation tool should allow us to analyze two or more different parallel processes, which use the same resources. Such analysis allows us to determine and evaluate the losses arising from the competition of different BP and to make a solution to reduce these losses.

As presented in (Ouyang et al. 2010) number of BP modelling approaches, like BPMN (OMG 2009), IDEF0 (1981), ARIS (Scheer and Nüttgens 2000), etc., provide different graphical notations for specifying resources associated with a BP. Our research is however focuses on developing a resource model that will be suitable for rule-based dynamic BP simulation.

Authors of (Ouyang et al. 2010) propose their resource classification object role model. It allows multiple resource assignment to the same activity, resource calendar definition and resource utilization logging. However, as can be seen from their model, authors of (Ouyang et al. 2010) do not talk about other attributes, which can be defined for resources.

The next thing we analyze in this paper is the implementation of the resource models in different BP management systems. In BonitaBPM (2015), which is developed to create process-driven business applications with a separation between business logic, business data objects and user interfaces, it is possible to model resources. However, the main drawback is that it is impossible to define additional attributes to the resources. For example, in BonitaBPM we can define that our company has ten cars. However, all those cars can differ by tjeir capacity, and one car can be used to deliver smaller quantity of items then another. Therefore, we have to choose a car with suitable capacity to deliver items at once.

In Bizagi (2015) a resource is defined as a person, equipment, or space necessary for the execution of a specific activity (by Bizagi “task”). Time and cost are the main constraints for processing activities. Therefore, the performance measures can be evaluated in Bizagi as follows: Sub- or over-utilization of resources; total resources costs; total activity costs; delays (time an activity waits for a resource); and a more accurate expected cycle time. However, it isn’t possible to define other attributes, showing different characteristics, for the resource.

In the next section we present a resource model for the rule-based dynamic BP simulation.

3 A RESOURCE MODEL FOR THE RULE-BASED DYNAMIC BP MODELLING AND SIMULATION

According to the related works, the authors of this paper present the requirements for the resource model as follows:

1. Resources should be linked with activities. An activity may have several resources linked with.
2. Resources should have unlimited number of attributes, which define different characteristics of the resource.
3. Resources should have a usage cost and a quantity.
4. After the end of activity a reusable resource should be stated as a free and returned to the store.
5. It should be able to get information about the current state of resources at any time.
6. Resources have to maintain the competitiveness of BP.

According to the presented requirement for resource model, the resource management data base model is presented in Figure 1 as follows.

The proposed resource management data base model (Figure 1) consists of two parts. The first part describes attributes of resources. As was described in the requirements, the resource attributes could be defined are as follows: cost, data period when resource could be used, which is important if a resource has the data of expiration, quantity and maximum possible quantity, which is important when the storage capacity in warehouse is limited. According to the quantity and maximum possible quantity it is possible to calculate is it necessary to supplement a warehouse or not. Each resource has a role. For example, a resource “Jon Jons” has a role “driver” and this role will be assigned as “human resource” accordingly. Each resource attribute should have a value with units of measurements, for example, “weight 1000 kg”. Table 1 presents an example of resource instances.
Figure 1: The Resource Management Data Base Model

Table 1: View of “Peugeot Boxer” Resource Attributes

<table>
<thead>
<tr>
<th>Name of the resource</th>
<th>Type</th>
<th>Attribute</th>
<th>Value</th>
<th>Unit of measurements</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peugeot Boxer</td>
<td>vehicle</td>
<td>volume</td>
<td>12</td>
<td>m³</td>
<td>2</td>
</tr>
<tr>
<td>Peugeot Boxer</td>
<td>vehicle</td>
<td>category</td>
<td>B</td>
<td>Null</td>
<td>2</td>
</tr>
<tr>
<td>Peugeot Boxer</td>
<td>vehicle</td>
<td>fuel</td>
<td>diesel</td>
<td>Null</td>
<td>2</td>
</tr>
<tr>
<td>Peugeot Boxer</td>
<td>vehicle</td>
<td>engine</td>
<td>1,6</td>
<td>Null</td>
<td>2</td>
</tr>
<tr>
<td>Peugeot Boxer</td>
<td>vehicle</td>
<td>years</td>
<td>2002</td>
<td>Null</td>
<td>2</td>
</tr>
<tr>
<td>Peugeot Boxer</td>
<td>vehicle</td>
<td>color</td>
<td>black</td>
<td>Null</td>
<td>2</td>
</tr>
</tbody>
</table>

Each resource type has standard number of attributes; however, if an additional attribute is necessary, it can be created for this resource. According to the attribute values the necessary resource, for example, a vehicle with category B, will be selected to execute an activity.

The second part of the model describes the usage resources in and linking resources within activities. As can be seen from Figure 1, a resource is linked with an activity, and an activity is linked with a process. Such linking allows us to determine what activities are executed within the process and what resources are used in those activities. Also it is necessary to specify when a resource is linked with an activity. All resources, which have “linked_from” data and have no “used_till” data, are busy. Only if “used_till” data is specified, resource is considered to be free or used. An activity can be linked with a resource, which in turn is used by other resource. For example, “Jon Jons” is using a resource “Peugeot Boxer” to deliver an item. Other attribute to be specified, when a resource is linked with an activity, is a quantity. If reservation of a resource is cancelled, then a resource is returned to the warehouse. Table 2 presents an example of a resource linking with an activity.
Table 2. An Example of a Resource Linking With an Activity

<table>
<thead>
<tr>
<th>Name of a process</th>
<th>Name of an activity</th>
<th>Linked from</th>
<th>Used till</th>
<th>Name of a resource</th>
<th>Resource of resource</th>
</tr>
</thead>
<tbody>
<tr>
<td>Product delivery</td>
<td>Prepare product</td>
<td>2015-04-16</td>
<td>2015-04-16</td>
<td>Jon</td>
<td>Null</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12.23</td>
<td>16.12</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product delivery</td>
<td>Load product</td>
<td>2015-04-16</td>
<td>2015-05-17</td>
<td>Peter Peters</td>
<td>Null</td>
</tr>
<tr>
<td></td>
<td></td>
<td>16.12</td>
<td>09.00</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Product delivery</td>
<td>Delivery product</td>
<td>2015-05-17</td>
<td>2015-05-17</td>
<td>Peugeot Boxer</td>
<td>Peter Peters</td>
</tr>
<tr>
<td></td>
<td></td>
<td>09.00</td>
<td>12.00</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

As can be seen from Table 2, we can determine the duration time and cost of each activity.

4 IMPLEMENTATION OF THE PROPOSED RESOURCE MODEL

The proposed resource model was implemented with Microsoft SQL Server (2015), C# (2015) and LingToSql (2015) library is used to implement connection between the resource database and rule-based dynamic BP simulation tool. For the simulation we used an ordering system of a middle-sized Lithuanian enterprise.

Below in Figure 2 we present an example of a function ALLOCATE, which is used to link a resource with an activity “Perform Order”. According to the defined parameters in a function ALLOCATE(Name of a resource, Resource-to-resource, Quantity, Duration), it finds the resource, which meets the parameters, and makes an insertion into the database of a resource link with an activity and the time when the resource was assigned to the activity.

Below in Figure 3 we present an example of a function RELEASE, which is used to free a resource from its linkage with an activity. According to the defined parameters in a function RELEASE(Name of a resource, Duration), it finds a record about linking a resource with an activity and set it free from the assignment by inserting a resource usage end time into database.

In Figure 4 we present an example of the rule-based dynamic BP simulation.

The description of Figure 4 is presented as follows. Process execution model section presents the graph of the simulated process instance. Activities presented in yellow are executed in this process instance simulation, activities in grey were simulated earlier in previous process instances, while activities in green were simulated just now. Process context section presents process context, including state of resources from the implemented resource model, and its change during execution of activities. Process context can change after execution of each activity. Simulation log is located below the process context.

5 CONCLUSIONS

The analysis of the resource perspective shows that the resource modelling is a non-trivial task, since in some cases it is difficult to predict their usage and linking with activities.

According to the related works the requirements for the resource model were defined and according to these requirements the resource model for rule-based dynamic BP simulation is proposed. For the implementation of the proposed resource model a resource management database model was created and implemented into rule-based dynamic BP simulation prototype.

The results obtained from the simulation prototype shows that the proposed resource model is suitable for rule-based dynamic BP simulation.
Acknowledgements. This work has been supported by the European Social Fund project Nr. 2013/0012/1DP/1.1.1.2.0/13/APIA/VIAA/051. The authors would also like to thank personnel of Information Systems Research Laboratory (ISML, 2015) for participating in the rule-based dynamic BP simulation system prototype development.

REFERENCES


WEB REFERENCES


ISML. 2015. Information Systems Research Laboratory, Institute of Applied Computer Science, Faculty of Fundamental Sciences, Vilnius Gediminas Technical University.
AUDRIUS RIMA is a Doctoral Student at Vilnius Gediminas Technical University. He received the MSc degree in Informatics from Klaipeda University in 2007. His research areas are as follows: business process modeling and simulation, resources modeling, use of resources in a concurrency manner.

ILZIE BIRZNIECE, Dr.sc.ing., is a researcher and lecturer in computer science field, and director of Study Department at Riga Technical University, Latvia. She has been participating in international scientific society by means of publications, scientific projects and conference organization committees. Research interests: machine learning, data mining and educational data analysis.

PETERIS RUDZAJIS, Dr.sc.ing, works as a researcher at Faculty of Computer Science and Information Technology in Institute of Applied Computer Systems. He is author and co-author of more than 20 papers in the field of information systems development. Research interests: information service systems development and conceptual modelling, information retrieval and extraction.

OLEGAS VASILECAS, prof. dr., is full professor at the Information Systems Department, principal researcher and head of Information Systems Research Laboratory in Vilnius Gediminas Technical University. He is author and co-author of more than 375 research papers and 5 books in the field of information systems development. His research interests include knowledge, represented by business rule and ontology, dynamic business process and information systems development. He delivered lectures in many European universities including London, Barcelona, Athens, Riga and Ljubljana. O. Vasilecas carried out an apprenticeship in Germany, Holland, China, and last time in Latvia and Slovenia universities. He supervised 11 successfully defended doctoral theses and now is supervising 4 more doctoral students. He was leader of number of international and local projects. Last time he leded „Business Rules Solutions for Information Systems Development (VeTIS)” project carried out under High Technology Development Program of Lithuania.

DIANA KALIBATIENE, assoc. prof. dr., is full time assoc. prof. at the Department of Information Systems, and researcher at the Information Systems Research Laboratory in Vilnius Gediminas Technical University. She participated in the High Technology Development Program Project “Business Rules Solutions for Information Systems Development (VeTIS)”. She is the member of the European Committee and Lithuanian Government supported SOCRATES/ERASMUS Thematic Network project “Doctoral Education in Computing” (ETN DEC), “Teaching, Research, Innovation in Computing Education” (ETN TRICE) and “Future Education and Training in Computing: How to Support Learning at Anytime Anywhere” (ETN FETCH). She is author and co-author of more than 30 papers and 1 book in field of information systems development. Research interests: business rules and ontology based information systems development, conceptual modelling and rule based dynamic business process modelling and simulation.
Flat vs Nested Firm Real-time Transactions : an Analysis of their Success Ratio

Mourad Kadess1,2,3, Laurent Amantm2, Majed Abdouli1,3, Mohailou Aleiyouka4, Alexandre Berred4
MIR@CL1, Pôle Technologique de Sfax BP 242, 3021, Sfax University, Tunisia.
LITIS2, Le Havre University, 25 rue Philipe Lebon 76600, Le Havre, France
CIFT3, Jeddah University, Saudi Arabia Kingdom
LMAH4, Le Havre University, 25 rue Philipe Lebon 76600, Le Havre, France
email: 1,3firstname.lastname@gmail.com, 2,4firstname.lastname@univ-lehavre.fr

KEYWORDS
Real-time databases system, Flat transactions, Nested transactions, Schedule, Monte-carlo Simulation

ABSTRACT

Real-time database systems are the most appropriate systems to manage applications which handle a large number of data and to provide real-time services. In fact, real-time database system guarantee the database logical consistency, on one hand, and it schedule the transactions in order to meet their deadlines, on the other hand. Hence, different scheduling policies are proposed to achieve these goals, e.g., EDF, AEDF, AEVD, APP, AEDF-CO, GEDF. However, the majority of studies are focused on the analysis of flat transactions behaviour. In this paper, we will study the scheduling of different transactions, specifically flat and nested transactions, using Generalized Earliest Deadline First (GEDF). GEDF is a protocol, in which transaction priority is assigned according to both deadline and the importance of transactions in the system. The improvement of GEDF scheduling policy and the influence of transaction structure on the system performances are investigated. Moreover, we analyse the influence on the success ratio of the execution capacity. This study enabled us to describe the behaviour of the different transaction success ratio.

Introduction

The main policy used to schedule transactions in RTDBS is Earliest Deadline First (EDF) which is based on a priority assignment according to the deadlines, i.e., the shortest is the transaction deadline, the highest is its priority. However, it is well known that EDF is not efficient in scheduling transactions in overload conditions, leading to the degradation of the system performances. Hence, different scheduling policies are proposed to overcome these drawbacks. e.g. AED (Haritsa et al. 1991), AEVD (Pang et al. 1992), APP (Datta et al. 1996), GEDF (Semghouni et al. 2007) (Kadess et al. 2013), AEDF-Co (Han et al. 2012), DS-EDF (Ramamritham et al. 2014). In the following, we present a brief description of the recent protocols.

(Han et al. 2012) have proposed a scheduling algorithm called Adaptive Earliest Deadline First Co-Scheduling (AEDF-Co). The performance goal of AEDF-Co is to determine a schedule for given sets of periodic user and update transactions such that the deadline constraints of all user transactions are satisfied and to maximize the quality of data (QoD) o’ the real-time data objects at the same time. (Ramamritham et al. 2014) have proposed DS-EDF to scheduling update transaction in order to outperform schedulability and minimize CPU workload. (Semghouni et al. 2007) and (Kadess et al. 2013) have proposed GEDF, a scheduling policy, which is based on a weight technique, i.e. a weight is assigned to a transaction according to its importance in the system. In (Kadess et al. 2014) authors are interested in analysing the performance of the system using nested transactions. In fact, the extended transactions are most suited for supporting the new real-time applications. In this paper, we propose to improve these studies. To this end, we compare the influence of the most important parameters, e.g., system priority, Execution capacity; on system performance under GEDF and EDF policies using flat and nested transactions.

The remainder of this paper is organized as follows. Section 2, presents briefly transaction and data models and the GEDF policy. Section 3 is devoted to the Monte Carlo simulation experiments. We show the influence of the transaction structure and the execution capacity. Section 4 concludes the paper and discuss some aspects of our future work.

System model

We consider only firm real-time transactions, i.e., late transactions are considered useless. Transactions are classified into update and user transactions. Update transactions are periodic and only write temporal data which capture the continuously state changing environment. We assume that an update transaction is responsible for updating a single temporal data item in the system. Each real-time data item is updated following a more-less approach where the period of an update transaction is assigned to be more than half of the validity interval of the real-time data (Xiong and Ramamritham 2004). Two types of user transactions are handled: flat and nested transactions. User transactions access at different data (real-time data and non real-time data). The num-

42
ber of operations generated for each user transaction follows a Poisson distribution. Data accessed by the operations of the transaction are randomly generated and built according to the level of data conflicts. User transactions are submitted to the system following a Poisson process with an average rate $\lambda$, into the active queue. For nested transaction, only the leaf sub-transactions can read or write non-real-time data and read real-time data. All the sub-transactions arrive in the system at the same time as general user transactions. The number of operations generated for each user transaction is uniformly distributed among the leaf sub-transactions. A transaction is considered committed if all its sub-transactions have already committed.

**GEDEF protocol**

GEDEF is a dynamic scheduling policy, in which the transaction priority is assigned according to both the deadline, which expresses the criticality of time and the SPriority, which expresses the importance of the transaction. We consider that the zero value of the Priority ($Priority = 0$) corresponds to the highest priority in the system. Transaction $T$ is assigned a priority by the formula:

$$Priority(T) = (1-a) \times Deadline(T) + a \times SPriority(T)$$ (1)

where:

- $SPriority$ : System priority is a parameter related to each transaction. It expresses the degree of importance of the task(s) executed by a transaction and defines its rank among all the transactions in the system. Two weight functions are used according to the transaction class to assign the SPriority value and are described in the following.

- $0 \leq a \leq 1$, is the weight given to $SPriority$ in the priority formula.

1- Update transactions class Let $MaxPeriode_T$ be the longest period among the periods of update transactions. The SPriority of an update transaction $T$ is computed according to the following formula:

$$SPriority_{update} = N \times \frac{Periode_T}{MaxPeriode_T}$$ (2)

2- User transactions class The user transaction importance $SPriority$ uses criteria based on both the transaction “write” set operations and the transaction “read” set operations. A user transaction $T$ is assigned a SPriority value by the following formula:

$$SPriority_{user} = MaxValue - \gamma \times Weight_T - (1-\gamma) \times DBA_{Value}$$ (3)

where:

1. $Weight_T$ denotes the weight of the current user transaction

<table>
<thead>
<tr>
<th>Notation</th>
<th>Signification</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\lambda$</td>
<td>User transaction arrival rate.</td>
<td>0.1 to 2.0.</td>
</tr>
<tr>
<td>Time</td>
<td>Duration of one experiment.</td>
<td>1000 clock cycles.</td>
</tr>
<tr>
<td>DBSize</td>
<td>Number of data in the DB.</td>
<td>300.</td>
</tr>
<tr>
<td>TD-size</td>
<td>Number of real-time data in the DB.</td>
<td>10%, 15% and 20% × DBSize</td>
</tr>
<tr>
<td>Min_{avi}</td>
<td>Minimal and maximal avi.</td>
<td>Min_{avi}=5 clock cycles, Max_{avi}=100 clock cycles.</td>
</tr>
</tbody>
</table>

2. $\gamma \in [0, 1]$ is the rank assigned to the transaction weight in the SPriority formula (see Table 2).

3. $DBA_{Value}$ is a uniform random variable between 0 and $(MaxValue – N)$

**System performance metrics**

To measure the system performances, we consider transaction success ratio and user transaction success ratio as metrics.

- The success ratio is given by:

$$SRatio = \frac{CommitT}{SubmittedT}$$

- The success ratio of user transactions:

$$SRatio_{User} = \frac{CommitT_{User}}{SubmittedT_{User}}$$

where $CommitT$ (respectively $CommitT_{User}$) indicates the number of transactions committed by their deadlines, and $SubmittedT$ (respectively $CommitT_{User}$) indicates all submitted transactions to the system in the sampling period.

In order to enhance this studies and describe a complete behaviour of success ratio, we give a reasonable approximation of user transactions success ratio $SRatio_{User}$ frequency distribution according to the system load. To this end, we follow theses steps:

We are reminded that a random variable $X$ follows a standard Beta distribution with parameters $p$ and $q$ if its probability density is given by:

$$f(x, p, q) = \frac{x^{p-1} (1-x)^{q-1}}{\mathcal{B}(p, q)}, \quad x \in [0, 1],$$

where:

$$\mathcal{B}(p, q) = \int_0^1 x^{p-1} (1-x)^{q-1} dx.$$
Table 2: Simulation parameters used for transaction characteristics.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>UserInterval</td>
<td>User transaction size.</td>
<td>[1, 51] combined operations generated by Poisson distribution.</td>
</tr>
<tr>
<td>NStr</td>
<td>Number of sub-transactions.</td>
<td>$N_{Str} = 4$.</td>
</tr>
<tr>
<td>UpdateSize</td>
<td>Number of operations in update transaction.</td>
<td>1 write operation.</td>
</tr>
<tr>
<td>D–U$pT$</td>
<td>Deadline of an update transaction (more-less approach).</td>
<td>$D_{–U}pT = \frac{1}{3} \times av_i$.</td>
</tr>
<tr>
<td>P–U$pT$</td>
<td>Period of an update transaction (more-less approach).</td>
<td>$P_{–U}pT = \frac{2}{3} \times av_i$.</td>
</tr>
<tr>
<td>SPriority</td>
<td>Intervals of SPriority.</td>
<td>$SPriority_{U}d = [0, 16]$, $SPriority_{U}ser = [16, 80]$.</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>Initialization of $\gamma$.</td>
<td>$\gamma = 0.8$.</td>
</tr>
</tbody>
</table>

By using the moment method, we estimate the parameters $p$ and $q$ by:

$$
\hat{p} = \bar{x} \left[ \frac{(1 - \bar{x})}{s^2} - 1 \right] \quad \hat{q} = (1 - \bar{x}) \left[ \frac{(1 - \bar{x})}{s^2} - 1 \right]
$$

where $\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$ is the sample mean and $s^2 = \frac{1}{n} \sum_{i=1}^{n} (x_i - \bar{x})^2$ is the sample variance.

Simulations and Results

We carried out Monte Carlo simulations which allowed us to study the transactions’ success ratio behaviour and the system quality of service. According to the system parameters given in Tables 1 and 2, the experiment was repeated 1000 times in each simulation in order to obtain a sample of 1000 values for the performances.

Influence of SPriority weight on system performances according to the system workload under flat and nested transactions

To show the influence of SPriority weight, we have assigned different values to the $a$ parameter under various system workloads. We note that the value of $\gamma$ parameter is set to $\frac{1}{3}$ in order to minimize the influence of DBA, i.e., database administrator interaction, in the SPriority Formula 3. We have used flat transactions in the first set of simulation and nested transactions in the Second set. The number of operations by transaction is included in [1, 51] and follows the Poisson distribution. For nested transactions, we have fixed the number of sub-transactions to 4.

Figure 1: Influence of the SPriority weight parameter according to system workload.

Figure 1 shows that independently of structure of transaction, system gives better performances under EDF at the beginning and this situation is reversed with increasing load. We note that the inflection point is reached rapidly with flat transactions, although we have used the same number of Flat and nested transactions and transactions have the same number of operations. Indeed, figures 1(a) and 1(a) show that the inflection point is obtained when $[0.8, 1.0]$ (respectively $[1.2, 1.4]$) with flat transactions (respectively nested transactions). This can be explained by the fact that the flat transactions are wholly restarted when data conflicts occur, however, only a subset of nested transactions are restarted when using nested transaction.

Moreover, system performances i.e., $S_{Ratio}$ and $S_{RatioUser}$ obtained with nested transactions outperforms the ones obtained with flat transactions under different values of SPriority as shown in Figures 2(a) and 2(b). However, this performance improvement is more important with $EDF$ than $GEDF_{a=0.3}$ and $GEDF_{a=0.5}$. Indeed, the improvement of $S_{Ratio}$ using nested transactions reaches 17% under $EDF$ and 13% under $GEDF_{a=0.3}$ and 11% under $GEDF_{a=0.5}$. In the same way, the improvement of $S_{RatioUser}$ is more important with $EDF$. It is close to zero when $\lambda \in [0.1, 0.5]$ then it increases rapidly to reach 48% when $\lambda = 1.3$. With $GEDF_{a=0.5}$ (respectively $GEDF_{a=0.3}$) this improvement of
Figure 2: Influence of user transactions structure on system performance.

\( \text{Sratio}_{\text{User}} \) doesn’t exceed 30% (respectively 36%). Figure 3 shows that the frequency distribution of the success ratio could reasonably be approximated by a Beta probability density function, according to the Kolmogorov-Smirnov (K.S.) statistical test.

Influence of computing capacity

The computing capacity is one of the most influential factors on the transaction success ratio. Hence, we analyse in this section, the influence of computing capacity on system performance with different transaction structures and percentage of real-time data under EDF and GEDF. To this end, we repeated previous experiences with various commuting capacities (Quantum = 10, 20 and 30).

In previous sections, we have shown that \( \text{Sratio}_{\text{User}} \) obtained with nested transactions outperforms the ones obtained with flat transactions independently of Spriority. In this section, we have carried out the identical experiences with different execution capacities. Results obtained, as shown in Figure 4 confirms that the \( \text{Sratio}_{\text{User}} \) obtained with nested transactions outperforms the ones obtained with flat transactions under different computing capacities. Moreover, we can note that the influence of structure is more important under EDF than under GEDF regardless of computing capacities. However, this influence decreases when the percentage of real-time data decreases. e.g. with EDF the improvement reach 17% and 11% with \( \text{GEDF}_{\text{E}-0.05} \) when the Quantum = 30 and when the Quantum = 10 the improvement don’t exceed 5% with EDF and 3% with \( \text{GEDF}_{\text{E}-0.05} \).

In order to enhance our analysis, we have varied the percentage of real-time data with different computing capacities. Results obtained confirms that when the percentage of real-time data decrease the \( \text{Sratio}_{\text{User}} \) increases with different computing capacities. Moreover, the influence of the database composition is less important when the computing capacity decreases, e.g. when the percentage of real-time data decreases from 20% to 15% the enhancement of \( \text{Sratio}_{\text{User}} \) reaches 12% when the Quantum = 20 but it doesn’t exceed 7% when the Quantum = 30 under EDF. This enhancement of \( \text{Sratio}_{\text{User}} \) with \( \text{GEDF}_{\text{E}-0.05} \) reaches 9% when the Quantum = 20 but it doesn’t exceed 5% when the Quantum = 30.

Figure 5 shows that the frequency distribution of the user transaction success ratio can be reasonably approximated by a Beta probability density function, according to the Kolmogorov-Smirnov (K.S.) statistical test regardless of execution capacities.

Conclusion

GEDF protocol is a multi-parameters scheduling policy based on the well-known EDF policy. Its parameters have to be tuned by taking into account the systems characteristics. Simulations conducted have shown that using nested transactions enhances the performance of the system. Nested
transactions performances are less sensitive than flat transactions to the parameters change. Moreover, GEDF is also less sensitive than EDF to the database composition and execution capacity. Furthermore, we have shown that the frequency distribution of the success ratio for user transactions can be approximated by a Beta distribution independently of the scheduling protocol used, the transactions structure and the execution capacity. Hence, we can predict the success ratio of the system without conducting further simulations, that often require a large computation time. In a future work, we plan to adapt the GEDF protocol to scheduling the specific transactions of MANETs and HETNET network.

REFERENCES


Discrete Event Simulation to Reduce the Effect of Uncertainties on Project Planning

Parminder Singh Kang1, Adam Aboutaleb1, Tim Embley2, James Glenn3, Colin Adams3 and Alistair Duffy1
1. Lean Engineering Research Group, Faculty of Technology, De Montfort University, Leicester, LE1 9BH, UK
Email: pkang@dmu.ac.uk, 0044 116 207 8089
2. Costain Ld. Costain Business Park, Maidenhead, SL6 4UB
3. Loxius Ltd. 2 Boulton Road, Southsea, PO5 1NT

KEYWORDS
Discrete Event Simulation, Risk Based Project Planning, Autonomous Modelling, Uncertainty, Optimization.

ABSTRACT
Planning is a vital decision making activity that influences the future of an organization by determining what tasks are to be performed, who required resources are and in what sequence. Organizations often follow a rigorous process to plan and deliver projects optimally based on the given resource and budget constraints. However, uncertainties increase the complexity of the planning process and contribute towards increased cost, delays and resource allocation issues. Therefore, it is important to understand the uncertainties and constraints associated with project activities and their effect on both business processes and organizational goals. Understanding the causal relationships between activities and constraints could allow organizations to operate more effectively and efficiently even in uncertain environments and lead to a more informed decision making process. This paper exemplifies the use of discrete event simulation tool to develop a strategically focused project delivery plan founded on the assessment of uncertainties that could arise during the delivery of the project. Proposed methodology follows a structured and systematic approach in order to identify the factors that can affect the delivery of the project and evaluate solutions that may mitigate or reduce the risk to As Low As Reasonably Practicable (ALARP). The main objective is to complement the existing project planning activities rather than replace the existing tools.

1. INTRODUCTION
The main aim of this research project is to provide a systematic approach to project planning that could deal with both complex aspects of the problem and uncertainties at different levels of project management. This paper illustrates one of the important aspects of the research project (AMDT – Funded by InnovateUK), which is project planning by using discrete event simulation (DES) and design of experiments (DoE) in order to better assess how the uncertainties and constraints might affect the project delivery. The real thrust behind this project is equipping the organizations with improved tools and techniques to support the decision making process. Also, it is utmost important due to the high level of competition and dynamic nature of the real world (Horrelan and Leus 2005 and Kumar and Phrommathe 2005). Going forward, the importance of effective project planning will be more important due to the new government regulations, for instance, according to the “Construction 2025” vision, the UK construction industry should deliver projects at lower cost (33%), lower emissions (50%) and faster (50%) (HM Government 2013). Therefore, there is a compelling case to develop project plans that are strategically focused and align with both organizational and other regulatory body’s objectives. According to (HM Government 2013 and Pich et al. 2002), there are several factors, which contribute towards the failure of a plan. For instance,
1) A project plan is developed based on planner’s experience, whereas the constraints and dynamic nature of real world change from project to project.
2) Lack of understanding about the project activities and communication between the stakeholders.
3) Information available to the project team has been often inadequate and doesn’t represent the complete system state, this introduces risks that can result in project failure.

In order to overcome these issues, the proposed approach integrates DES and DoE in order to understand the causal relationships that could potentially affect the project delivery. This can assist decision makers to make more informed decisions at strategic, tactical and operational levels and select the project with more realistic delivery deadlines and support daily operations respectively. The paper is organized as; Section II exemplifies the concept of DES and project planning, which forms the main part of the literature review. Further, Section III and IV provide the problem definition and illustrated the proposed methodology respectively. Section V, includes the case study based on data provided by Costain Ltd. Finally, Section VI and VII present the results and the conclusion respectively.

2. DES AND PROJECT PLANNING
2.1 Project Planning and Uncertainties
Planning is the process of determining what to do, how to do it and what are the resource requirements to deliver the plan according to deadlines, but in most cases by using the diverse resources and limited budget (Mantel et al. 2005). According to (Litman 2013), the role of planning is to create a logical and systematic decision-making process that results in the best actions. Good planning follows a systematic process that builds on the planning principles by considering the underlying causes of a decision, not just from a single perspective. A good plan exhibits the following properties (Litman 2013 and TBS Report 2005);
1) Comprehensive; analysis of factors affecting the plan.
2) Efficient; exclusion of non-value added activities.
3) Inclusive; stakeholders should be included in the process.
4) Informative; results are understood by stakeholders and decisions are backed up through relevant information.
5) Integrated; must include both long and short term goals.
6) Logical; each step is based on a systematic sequence.
7) Transparent; improved understating of business process.

Despite following best planning practices, in most cases projects fall behind the schedule during the implementation phase due to the issues arose during the delivery of the plan (Litman 2013 and TBS Report 2005). It is mainly due to the variability and randomness endorsed by real world, causal relationships due to the process-process (PP) and process-constraint (PC) interrelationships and lack of available information and visualization capabilities.

This leads towards the idea of risk based planning by considering the PP and PC interactions and providing a What-If analysis before reaching to the final decision in order to increase the value-added activities, especially in multi-project environments. Numerous approaches have been proposed by researchers, for instance Hans et al. (2007) has proposed a positioning framework to support decision making at different levels of project planning and to manage the uncertainties for the multi-project planning environment. Anavi-Isakov and Golany (2003) has used the concept of constant work-in-progress (CONWIP) to manage the number of active projects in a multi-project environment. Pitch et al. (2002) conceptualized the project management approach as a payoff function depends on the state of the world and the chosen actions. The complexity of the payoff function depends on the adequacy of available information, as the available information affects the decisions made by the project team and hence, the state of the world.

2.2 Discrete Event Simulation

DES is the process of codifying the behavior of a complex system as an ordered sequence of well-defined discrete sequence of events (McGregor and Cain 2009). The DES concept can be explained by using three basic elements of simulation modelling, which are (Maria, 1997);

1) System; represents the real life, e.g. manufacturing plant, bridge construction, patience flow etc.

2) Model; provides a simplified representation of some system of interest at given time or space to improve the understanding of real world (system).

3) Simulation; is the operation of a model of the system, when operated in its’ time or space in order to understand and manipulate interactions between system parts and as whole.

DES modelling approach is used to support the overall organizational decision making process by investigating system in a systematic and structure manner. The most common examples are, for validation before an existing system is altered or a new system built, to reduce the chances of failure to meet specifications, to eliminate unforeseen bottlenecks and uncertainties, to prevent under or over-utilization of resources, and to optimize system performance (Maria 1997). There are numerous examples, where DES or simulation based approaches have been successfully used within different sectors under various initiatives. For instance; understanding the production process dynamics for engine refurbishing plants for US Air Force to standardize and optimize the operations (Fass, 2009), to optimize the JIT production process for automotive component-manufacturing environment by investigating the effect of system constraints on production environment (Lummus 1995), to validate the future state for a Lean transformation process by including the time based random variability for different processes (Kang et al. 2010 and 2013) and to reduce the setup time for sheeting operation in pulp and paper manufacturer scenario (Sandanayke et al. 2008). The advantages of using simulation modeling are (Maria 1997, Sandanayke 2008, Marvel and Standridge 2009, Detty and Yingling 2000);

1) by modelling , and observing the system’s operation in detail over long periods of time a better understanding of the system of interest; 2) hypothesis testing; 3) investigating new operations, procedures, rules and flow; 4) simple illustration of complex systems; 5) the opportunity to exploit system constraints to measure the effect of variability on KPIs; 6) understanding of system constraints, bottlenecks and uncertainties; 7) business process optimization, 8) systematic problem solving approach and 9) training and visualization.

3. PROBLEM DEFINITION

According to (Hans et al. 2007), by 2018 $9 trillion will be spent globally on larger projects and UK is projected to spend over £250 billion on more than 500 major projects. However, larger projects have major issues due to the increased cost, delays, resource allocation, supplier engagement and other uncertainties. For instance, Cost overruns and significant delays for Euro tunnel between the UK and France, cargo railway link between Netherland and Germany and construction of Kuala Lumpur’s new international airport (Beckers et al. 2013). In most cases, the risks associated with one phase (process) are neglected without understanding their knock-on impact on another phase (process). From the practical perspective, it is utmost important that the new methodologies and tools need to be developed to provide the effective project management. Traditional and existing planning approaches have the capability of multi-project planning, however, these approaches use deterministic activity task durations and don’t support the What-If analysis to understand the causal relationships among the different activities and associates constraints (Hans et al. 2007, Payne 1995 and Loya et al. 2000). Also, most of the time project planning is carried out based on the historical information and planner’s experience, where (experienced) planner might have considered potential uncertainties in advance. However, it is impossible to consistently allocate the scarce resources at an optimal level, among different projects with a high degree of complexity and uncertainty by manually performing What-If analysis. Further issues are associated with lack of adequate information and understanding of causal relationships (Pitch et al. 2002). Therefore, the major problem is that the three basic dimensions of project success, i.e. time, quality and cost are often under risk, due to the lack of causal relationship modelling and adequate information (Litman 2013 and Hans et al. 2007). According to (Zilicus Solutions, 2012), the key elements of project planning are time, quality and resource, known as TOR constraints. Any uncertainties associated with TOR are a risk to the project scope and at the same time any change in the scope has a direct effect on either all or any of time, quality and resources of a given project.
4. PROPOSED METHODOLOGY

The proposed methodology integrates DES and DoE approach to investigate the effect of uncertainties on the project plan. The main focus of this section is on the DES modelling based on the project activities (Figure 1):

1. Select Project – The process starts with the selection of a project(s), which can be either at strategic level to generate a feasibility study, business case, current resource assessment, etc., or at operational level to support informed decision making for day-to-day operations by incorporating the real world uncertainties and constraints.

2. Business Process Mapping (BPM) – once the projects are selected data needs to be collected regarding:

   1) Scope and Objective; project scope and objective is the most important element to provide the direction for the project team throughout the delivery of the project i.e. what needs to be included in the plan, such as activities, time, resources, quality etc. Objectives provide a quantifiable way to achieve the organizational goals, for instance, maximise profit, reduce costs, improve quality, etc.

   2) Activities and Relationships; definition of activities that required for delivery of plan or final product. This also includes determining the activity timings, associated distributions, milestones and resources required w.r.t project timeline. Most importantly, the sequence of all activities needs to be defined, which can be either sequential, parallel or both.

   3) Activity Risks; careful consideration must be given to identify the real world constraints and uncertainties in order to identify any risks associated with TQR and scope. For instance, operational environment constraints, resource constraint such as skill level, number of resources available, budget constraints, etc. Along this, activity dependencies should be identified in order to understand activity-activity relationships under uncertainties.

   4) Data Collection; in order to generate the simulation model, data needs to be collected based on the previous steps. This data can be extracted from the historical data or meetings with stakeholders. In most cases, this may require several meetings with stakeholders in order to verify the information.

In general, BPM improves the understanding about the system under investigation. Also, it works as a holistic approach by improving the communication between the stakeholders.

3. DES and DoE – DES and DoE are generated based on the information collected through the BPM related to activities and risks.

1) DES Model Development; the main aim of this step is to conceptualize the activities to DES modelling elements (ME). An activity becomes a DES ME and activity time can be associated as a fixed number or a distribution. Whereas, DES scope and KPIs are derived from project scope and objectives. The main aim here is to generate the DES model similar to the real world, but simpler by capturing all the boundaries accurately. Project activities are represented in the form of MEs, as (McGregor and Cain, 2009);

Figure 1: Proposed DES Framework

- Work Entry Point/Source – represents a triggering event, it can be temporal or causal, where work to be done appears in simulation for the first time. For example, generation of order, information, material etc.,

- Activity/WorkCentre – represents a place where work takes place on a work item. For instance, sign documents, data collection, manufacture products, etc.,

- Queue – represents a place where work to be done can wait until appropriate Resources or Activities are available, such as documents waiting for signatures, items waiting for assembly, etc.,

- Resource – represents items in the simulation environment that are required at Activities in order for the Activity to be carried out on a Work Item. For instance, supervisor, floor manager, etc.

- Work Complete/Sink – represents a place where work that is complete, or otherwise finished, leaves your simulation. For example, completed products, project milestones, and the throughput as the number of projects completed.

Along ME, other simulation entities need to be defined, some of which might apply to the specific project. The common ones are;

- Simulation run time; for how long model needs to run. This is equivalent to the time period of project under investigation.

- Results collection period; this might be different from that of the simulation run time for some cases. The results collection period is how long the simulation should be run before automatically stopping while collecting results.
- **Work Item**: in this specific case, work item represents the system under investigation. Work item will go through the sequence of activities defined in the form of modelling elements.

- **Trials**: the main purpose of the trial is to check the reliability of results to mimic the real life scenarios with uncertainty. A trial allows a series of runs of the simulation performed with the same settings for all parameters other than “random numbers”. This provides more rounded results and improves accuracy in terms of proposed KPIs.

- **KPIs**: KPI needs to be selected based on the defined objectives during the BPM. DES (Simul8) provides a default set of KPIs associated with different ME, for instance, Throughput, %Waiting, %Working, %Blocked, %Stopped, Resource Utilization and Queue Size Queuing Time, etc.

2)**Generate DoE**: experiment set is developed using the activity risk information collated from the stakeholders and planners. Experiment designs are used intensively in different industrial scenarios by process engineers to understand the important cause and effect relationships and valuable process information (Czupryna and Rankin 2014 and Tanabe et al. 2014). Full factorial is used to formulate the experiments in order to study the PP and PC interactions based on the selected response variables (KPIs). Full factorial design allows user to test all the possible combinations i.e. the interactions between the factors and response variables. Going into details with respect to this section is out of scope of this paper. Then follows an iterative approach to verify and validate the DES model resulting in an accurate and robust project plan. The validation requires meetings with stakeholders and planners to support the acceptance and buy into the plan.

4). **Link DES and DoE**: once the DES model is validated, the next step is to automate the experimentation process by linking the DES and DoE in order to understand and examine the effect of variability on the system under investigation.

5). **Results Collection and Causal Analysis**: run simulation based on the defined simulation run time and results will be captured based on the set simulation run time or results collection period. Further, based on the causal analysis between the constraints and KPIs optimal plan is generated.

6). **Review and Publish the Plan**: once the optimal plan is generated based on the given constraints and uncertainties, the results are disseminated to stakeholders for review and approval prior to the plan being published in the central database. During the life of the project, if there is any change to project scope, KPI, constraints, risks and stakeholders then DES and DoE need to be updated in order to investigate the effect of these changes.

7). **Execute Plan**: includes the real file execution of the plan.

5). **CASE STUDY**

The case study presented in this paper targets the water sector at Costain Ltd., especially a set of projects Costain delivers for Severn Trent Water (STW). Costain aims to deliver projects in the Lean way by using most cost effective practices, but with no compromise on the quality of service.

1). **Select Project** — the maintenance and service projects that have shaped this case study are received from STW at the start of each fiscal year. These projects need to be completed throughout the year. From the strategic planning aspect all projects follow the same high-level activities. Therefore, strategic planning forms the main focus of this paper. Under the current practices, the plan is currently generated through several project meetings before the start of the fiscal year and constantly updated throughout the year to incorporate any changes. Proposed methodology was validated using the “Block Y3 programme”, which ran from April 2012 to March 2013. This one-year programme included 31 projects with high levels of variability and complexity. The aim here is to optimise the Y3 Programme in order to maximise the number of projects delivered per year (Throughput). Along this, the proposed methodology looked at maximising the resource utilization.

2). **BPM** — information regarding the project activities in Block Y3 was collected through meetings and discussions with the process owners. Project activities at strategic level can be given as;

1). **Scoping**: includes the activities related to the initial scope issued by STW to generate preliminary cost estimates, requirements. This includes visits to site by Costain team.

2). **Contractor Pricing**: once the scope is updated, Costain will start tendering for a subcontractor to carry out the work during the construction phase.

3). **AC/SD Engineering Review**: activities related to the submission of the final price and sign off the contract.

4). **Procurement**: purchase/lease of the required materials, equipment, etc.

5). **RAMS**: Carry out a risk assessment and method statement before commencing the work on site.

6). **Construction**: maintenance and construction activities. It is important to note that in this paper from the strategic viewpoint construction is considered as one activity.

7). **Handover**: preparation of the handover documentation and handover site to STW.

8). **ADM and COGNICA**: capitalization and asset management-related activities.

As shown in **Figure 2**, the Scoping, Contractor Pricing and

![Figure 2: Simple Model Representation](image-url)

*NOTE: Figure 2 illustrates the sub-activities (1.1, 1.2, etc.) w.r.t to the each activity identified above to highlight the resource requirement at sub-activity level*
AC/SD Engineering Review activities are sequential; while the subsequent stages are parallel with some activities are dependent on each other. Resources are assigned to an activity from the resource pool based on the available resource. The five main types of resource are: Delivery Manager, Site Manager, Site Supervisor, Site Foreman and Quantity Surveyor (QS). Once the variability of the project plan was defined and agreed upon with the process owners, it was translated into a simulation model with the main attributes as shown in Table 1. The simulation run time and results collection period were set as the Y3 block period (April 2012 to March 2013). The variability in such a model can come from different factors:

1) **Projects Planning**: The start and finish time of each activity and which resource to be assigned to each activity from the resource pool. This factor is represented in the simulation model by the inter-arrival time between work items (i.e. projects) at each activity.

2) **Available Number of Resources**: Number of resources available for each type, such as delivery manager, site manager, etc. During the resource allocation process the main aim is to make sure there is no resource conflict and utilization is maximized.

3) **Batch Size Leaving between Stages**: The number of projects to be completed in each stage before moving forward to the next stage. The DES model investigates four levels between the minimum and maximum batch sizes to maximize the throughput.

4) **Processing Time of each Stage/Activity**: Project activities vary in processing time between projects.

5) **Project Complexity**: The project complexity is measures based on the duration of the construction stage; the longer the construction time the higher the complexity index.

Based on above, first three variability factors represent the first level of the Activity Risks. These can be controlled and it is the main aim of DES to determine their optimal settings to reduce this risk; therefore these variables were referred to as “Controllable Factors”. In other words, in this case the boundary values and variability information is available and What-If analysis can be done using the DES model. Table 2 gives the list of controllable factors and the associated levels. The second level of Activity Risks is represented by the last two factors. Here the time constraints are imposed to the project plan where the risks associated with the uncertainty of the activity duration are defined within the simulation model.

As mentioned, the variability of these activities from one project to another is considerably high. Table 3 gives the accumulative variability for different activities within Y3 Block represented by the normal distribution. However, the simulation model used the individual timings for each project activity for better accuracy. Full factorial DoE was then used to generate a list of all possible combinations of manageable Risk scenarios that could occur in terms of the controllable factors and results were collected from the simulation model for two KPIs (throughput and resource utilization), which are presented in Section VI.

### RESULTS DISCUSSION

It is evident from figure 3, increasing the number of resources will improve the throughput to a certain point; after that it will just reduce the resource utilization% with a steady throughput. This point is considered the optimal point and the optimal project plan was investigated using the number of resources at optimal point. Further, Figure A1 and A2 (appendix) shows the impact of each variability factor on the two KPI’s (throughput and resource utilization). As shown, the increase of number of resources and batch size in general reduces the resource utilization but increases the throughput. Therefore, further investigation was done to determine the optimal number of resources. The idea here is to balance the controllable factors, according to the current state of variability in order to achieve maximum throughput and improved resource utilization.

For the optimized scenario, the simulation model studied the variability associated with the uncontrollable risks and then generated an optimized project plan with fixed activity durations to be compatible for potential integration with project planning software. The completed projects with optimized plan are 27 out of 31 projects with an increase of 3 projects (i.e. 10%) over the actual completed projects within Y3.

### Table 1: Project Plan to Simulation Model Translation

<table>
<thead>
<tr>
<th>Project Terminology</th>
<th>Plan Element</th>
<th>Simulation Model</th>
<th>Simulation Attribute</th>
</tr>
</thead>
<tbody>
<tr>
<td>Activity Duration</td>
<td>Activity</td>
<td>Activity</td>
<td>Processing Time</td>
</tr>
<tr>
<td>Activity Finish Time</td>
<td>Activity</td>
<td>Activity</td>
<td>Processing Time</td>
</tr>
<tr>
<td>Project Start Time</td>
<td>Work Item</td>
<td>Activity</td>
<td>Processing Time</td>
</tr>
<tr>
<td>Project Finish Time</td>
<td>Work Entry Point</td>
<td>Inter-arrival Time</td>
<td>Processing Time</td>
</tr>
<tr>
<td>Resource</td>
<td>Resource</td>
<td>Resource</td>
<td>Resource</td>
</tr>
</tbody>
</table>

### Table 2: Controllable Factors and Levels

<table>
<thead>
<tr>
<th>Controllable Factor</th>
<th>Level 1</th>
<th>Level 2</th>
<th>Level 3</th>
<th>Level 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Resource - Delivery Manager</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Resource - Site Manager</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Resource - Site Supervisor</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Resource – Site Foreman</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Resource – Quantity Surveyor</td>
<td>1</td>
<td>2</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>Batch Size Leaving</td>
<td>1</td>
<td>8</td>
<td>16</td>
<td>31</td>
</tr>
</tbody>
</table>

### Table 3: Activity Timing Variability

<table>
<thead>
<tr>
<th>Activity Name</th>
<th>Attributes</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scoping</td>
<td>Avg. Processing Time</td>
<td>0.968 Days</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>0.180 Days</td>
</tr>
<tr>
<td>Contractor Pricing</td>
<td>Avg. Processing Time</td>
<td>8.285 Days</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>3.275 Days</td>
</tr>
<tr>
<td>AC/SD Engineering Review</td>
<td>Avg. Processing Time</td>
<td>13.452 Days</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>10.633 Days</td>
</tr>
<tr>
<td>Procurement</td>
<td>Avg. Processing Time</td>
<td>21.097 Days</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>15.441 Days</td>
</tr>
<tr>
<td>RAMS</td>
<td>Avg. Processing Time</td>
<td>8.935 Days</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>3.119 Days</td>
</tr>
<tr>
<td>Construction</td>
<td>Avg. Processing Time</td>
<td>13.323 Days</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>15.861 Days</td>
</tr>
<tr>
<td>COGNICA/ADM</td>
<td>Avg. Processing Time</td>
<td>59.387 Days</td>
</tr>
<tr>
<td></td>
<td>Standard Deviation</td>
<td>91.539 Days</td>
</tr>
</tbody>
</table>
7. CONCLUSION AND FUTURE WORK

Once the DES model is developed, proposed approach provides both proactive and reactive solutions to optimize the project plan. In the proactive manner, risk based assessment can assist an organization to understand how many projects with a given complexity level and acceptable level of risk are likely to be delivered using current resources and investigate about the future resource requirements. The key criterion of a proactive approach is to alleviate the consequences of uncertainties prior to the start of the project, e.g. by allocating the flexibility in a plan to uncertain activities, or to the periods where there are uncertainties. On the other hand, reactive approach is to make more informed decisions by providing the What-If analysis when activities go out of sync due to unforeseen circumstances. The main aim is generating the best possible reaction to a disturbance that cannot be absorbed by the plan without changing it. According to Hans et al. (2007), replanning is important to re-optimize or repair the complete plan after an unexpected event occurs. According to existing reactive approaches the impact to be absorbed by the slack in a plan. However, re-planning can benefit from the What-If analysis provided through proposed methodology instead of relying completely on planner’s experience. Along this, proposed approach could provide a number of benefits at all levels of project planning;

1) **DES;** following a systematic process to develop the DES model, which will improve the business process understanding and internal (organizational) and external (government, contractors, material supplier) constraints.

2) **What-If Analysis;** an inclusive approach to identify and understand the PP and PC interactions based on the given constraints and KPIs. Hence, informed decision making by focusing on the priority areas.

3) **Visualization;** visualization capabilities of DES can be used to communicate information effectively with stakeholders and at different levels within the organization. Therefore, proposed methodology can be seen as a consultative process that can help decision-makers to make informed decisions based on visual evidence about resource allocation and constraint settings supported through the results. Also, based on the three main benefits of the proposed methodology, it fulfills the requirements for a good project plan identified in literature review section (by Litman 2013 and TSB Report 2005). Further, proposed approach complies with three fundamental modern project management strategies;

1) **Instructionism;** planning based using the DES and DoE outputs, where both DES and DoE are result of BPM.

2) **Learning;** ability to re-plan or generate a completely new plan during the project to address the unforeseen circumstances. This is supported by both DES and DoE.

3) **Selectionism;** pursuit of multiple candidate solutions until the best can be identified; integrating DoE with DES. For future work, the planned activities are; 1) validate the proposed approach using other project planning case studies and 2) use evolutionary algorithms to validate the results and to determine the feasibility of finding a solution quicker than the full factorial (but closer to the best solution).

8. ACKNOWLEDGEMENT

The authors would like to thank the InnovateUK (Grant Reference NO: 18834-132285) and De Montfort University for funding this research and Costain Water Sector Team to provide the process information for model development and validation.

9. REFERENCES


10. Appendix - A

Figure A1: Impact of Factors on the Resource Utilization

Figure A2: Impact of Factors on the Throughput
INTELLIGENT DATA SIMULATION
METAHEURISTICS APPLIED TO THE AUTONOMOUS MOVEMENT OF INTELLIGENT AGENTS

Dâmaris da Silva Bento
Bárbara da Costa Rodrigues
Juliano Costa Penha Alves
Bruno Luan de Sousa
Angelo Batista Neves Júnior
Lindomar Matos de Paula
Alex F. V. Machado
Department of Computer Science
IFSEMG – Câmpus Rio Pomba
36180-000, Rio Pomba, Brazil
E-mail: damaris.s.bento@gmail.com

KEYWORDS
genetic algorithm, GRASP algorithm, simulation, machine learning

ABSTRACT

In the same way as technology develops, researchers, developers and companies have been trying to improve the intelligent agents and the robots, making them more similar to the natural movements. However, this development is a complicated task. Robots and intelligent agents require various settings of parameters, requiring a lot of effort, and also an expert in the subject. This article proposes the simulation of a three-legged intelligent agent which aims to make walking as natural as possible. For the purpose of it, two metaheuristics are being used: the Genetic Algorithm (GA) and the GRASP Algorithm. An autonomous simulator of movements in Unity 3D was created, and a comparative analysis between the two algorithms was made. As result, although the GRASP had a smaller memory consumption, GA generated better results and was superior in the CPU, GPU and system processing time tests.

INTRODUCTION

Nowadays there are several studies of robots with wheels, but these can not (or struggle to) go on land or "overcome" obstacles designed for humans, for example, stairs (Heinen and Osório 2006; Arakawa and Fukuda 1996; Fukuda et Al. 1997).

Following the technological evolution, researchers, developers and companies have sought to perfect the intelligent agents and robots, having them become more and more similar to natural movement (whether human or animal). According to Jennings (1994) an intelligent agent is a computer system located in an environment that is able to act with flexible autonomy aimed at achieving the objectives for which it was designed.

In robotics, companies such as Boston Dynamics, have shown robots that possess skills to get around naturally as humans even in rough terrain and they are used in several areas, mainly for military purposes in the US (Darpa). In the video game field it has been no different. Developers have sought to make the virtual world more like the real one, simulating scenarios and movements, especially the first-person games (action and shooters) and sports games (football and basketball), for example. These games allow characters to traverse regular and irregular ways, always simulating human movement.

The development of intelligent robots and agents with legs is a complicated task due to the need of setting various parameters. Feeding and/or setting these parameters require much effort and there is the need of a specialist. For this reason, it becomes interesting the use of artificial intelligence, which through machine learning, the agent may receive initial parameters and adapts to the middle until it learns how to perform the task alone (Heinen and Osório 2006).

A machine learning technique that best fits for this task is the use of Genetic Algorithms (GA). This becomes plausible because according to Darwin (1859), in his theory of evolution, locomotion mechanisms of various forms of life resulted from natural evolution. Moreover, because of the GA being biologically inspired, their use is a natural solution. The algorithmic solution through genetic algorithm is seen and indicated by Heinein and Osorio (2006) and Pacheco (1999).

According to Rodrigues (2004) and Sucupira (2004) a heuristic algorithm is one that does not have a complete mathematical solution about their behavior. However, the metaheuristics are sophisticated heuristic techniques, easy to implement and very efficient, which differentiates them from other heuristics. They are algorithmic structures that are applied to different optimization problems. The main ones being: GRASP (Feo and Resende 1995), Genetic Algorithm (Holland 1992), Busca Tabu (Glover 1977), Simulated Annealing (Kirkpartrick et al., 1983; Cerny, 1985).

This paper seeks to explore the following heuristics: Genetic Algorithm and GRASP (Generic Search Algorithm for the Satisfiability Problem), in a segment of artificial evolution, a simulator of the evolution of the walk of an intelligent agent, and will be a comparison of the performance of these two algorithms.
Autonomous Movement

An autonomous evolutionary movement simulator of an intelligent agent is a program about life in artificial evolution. Simple creatures formed by connection boxes and motors, computationally simulate evolution by means of mutations and selections that can be modified and adapted until they find the best way forward. For generations the creatures are undergoing changes, evolving the rudimentary ability to walk. This program makes possible a clearer and more comprehensive understanding of the evolution of a species. This process is completed in minutes.

The development of this simulator depends on the type of algorithm used in its implementation. In this work it was implemented in the Unity3D engine, using two artificial intelligence algorithms: Genetic Algorithm (GA) and GRASP, and it is possible to clearly see the evolution of creatures and a comparison between the performance of the two algorithms used will be done.

These two algorithms have been chosen because it is a complex problem with many variables, for they are optimization algorithms and because of the GA being more widespread in the area. Furthermore, GRASP’s modelling is based upon the GA.

Genetic Algorithm (GA)

Developed by Holland in the 70s, the Genetic Algorithm is an evolutionary algorithm inspired by evolutionary biology that uses techniques such as inheritance, mutation, natural selection and recombination. It is used in computer science to find approximate solutions to optimization (Pontes 2007).

It is implemented with a computer simulation, where evolution occurs when it is created an initial population with random solutions modeled on chromosomes. At every generation, these chromosomes are recombined and mutated to generate a new population, which will be the entrance to the next generation. The algorithm only ends when it is found the best solution, which will be defined by the fitness function (Loureiro 2005).

GRASP

GRASP (Greedy Randomized Adaptive Search Procedures) was initially described in the work of Feo and Resende (1995) and is essentially a combination of a constructive heuristic with a Local Search. GRASP is an interactive process that consists of two stages: the construction and the local search. When a feasible solution is found in the construction, it is applied a greedy function, then the local search algorithm. The best solution found is maintained as a result (Festa 2007).

Construction stage
A viable solution begins with an empty set and then continues by inserting elements one at a time. At each iteration the elements are assessed through a greedy function that puts the solutions in a list of top candidates called the shortlist. The probabilistic component of GRASP chooses one of these solutions on the shortlist randomly and adds it in the final solution (Feo and Resende 1995).

Local Search stage
The local search is performed on the shortlist that allows an improvement in each built solution. The algorithm works iteratively, replacing current solutions for better ones. It walks through the neighborhood to find an optimal place, but these are not always generated. The efficiency will be given by the neighborhood structure, the search strategy and the construction strategy (Feo and Resende 1995).

RELATED WORK

Arakawa and Fukuda (1996) and Fukuda et al. (1997) proposed in their work methods of stable movement generating for a biped robot. The first one used genetic algorithm to optimize the energy applied to the movement and the following used recurrent neural networks, which together with GA, enabled learning ability and adaptive mutation of self-operator. At the end of their work, the robot’s locomotion had become very similar to the natural movements.

Heinic and Osório (2007) did a study in robots with legs research field and the development of LegGen system, which is used for creating and managing stable walks for legged robots in a physically based simulation environment automatically. They used genetic algorithms to optimize the robot control parameters and the results showed that it is possible to generate stable walks using GA efficiently.

Capi et al. (2000) in his study proposed a solution to reduce energy consumed by a biped robot to walk. Genetic Algorithm was used to calculate the trajectories of angles that are extremely important for robot stability, allowing it to perform tasks such as going up and down stairs and overcome various obstacles requiring little power. Along with the GA it was also used neural networks.

There were no work related to bipedal locomotion using the GRASP (Greedy Randomized Adaptive Search Procedure).

Both works discussed in this chapter use the genetic algorithm, which was a motivation for the use of GA in our comparison. Since nothing was found related to the GRASP we chose it to be a differential and a possible solution to optimize the walk of intelligent agents.

The work of Capi uses GA to calculate the angle of movement of bipedal, this parameter was used in our project.

PROBLEM FORMULATION AND PROPOSED METHOD

Our work was designed and developed in the Unity3D engine, because of the ease of viewing the development and programming of our simulator. Furthermore, the
platform provides features that facilitate this simulation such as the environment in which the agent will walk, floor modeling and the physics that surround them.

We chose a smart tripod agent for believing it be easier to achieve good results because of its physical structure and for being a differential, since the experiments found were mostly bipedal intelligent agents and animals and hexapods. This does not mean that the algorithm cannot be used for other types of intelligent agents with legs.

Still, the smart tripod agent was used to accomplish the comparison between two types of artificial intelligence algorithms: Genetic Algorithm and GRASP.

In the Unity3D environment it was created a smart tripod agent that consists of 4 cylinders (three feet and head). The modeling of the problem was made by analyzing basic factors of the walk from one tripod agent. Among them, it was found relevant: the initial angle of each leg, the return movement of these legs, and the movement time of each leg. It was also analyzed the distance traveled by each agent. The definition of the parameters was made arbitrarily. Below is the definition of each parameter and Table 1 shows the range of values used by each.

Definition of parameters:

- **Shift (Right, Left, Middle):** this parameter allows the agent's leg rotate 360° around the head. With the value "-1" the rotation is done counter-clockwise and the value "1" it is done clockwise. The "0" locks the rotation of the leg.
- **Initial angle (Right, Left, Middle):** this parameter determines the degree of the angle that the agent's leg will start with (with a range that allows to rotate or zigzag).
- **Time (Right, Left, Middle):** is the movement time for each leg.
- **Agent distance:** distance traveled by the agent at the end of each generation.

Table 1 – Parameters adopted in the algorithms

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Value range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Right shift</td>
<td>-1 or 1</td>
</tr>
<tr>
<td>Left shift</td>
<td>-1 or 1</td>
</tr>
<tr>
<td>Middle shift</td>
<td>-1 or 1</td>
</tr>
<tr>
<td>Right Initial Angle</td>
<td>-140 to 140</td>
</tr>
<tr>
<td>Left Initial Angle</td>
<td>-140 to 140</td>
</tr>
<tr>
<td>Middle Initial Angle</td>
<td>-140 to 140</td>
</tr>
<tr>
<td>Right Time</td>
<td>0 to 77</td>
</tr>
<tr>
<td>Left Time</td>
<td>0 to 77</td>
</tr>
<tr>
<td>Middle Time</td>
<td>0 to 77</td>
</tr>
</tbody>
</table>

Figure 1 depicts the motion of a leg tripod, illustrating what happens to each of the legs during the execution of algorithms.

**SOLUTION THROUGH GENETIC ALGORITHM**

In the modeling of our Genetic Algorithm, it was used the tripod creature as chromosome where the parameters used were considered the genes. In the initial population, each gene of a chromosome is created randomly by choosing a particular value of a range of predefined values for each type of gene. After that, the crossover, two chromosomes are chosen randomly and exchange genes with each other. Then, the mutation step selects three genes and assigns random values to them. Completed these three steps, the chromosomes are stored in a vector that will be sorted according to the distance traveled by the agent (chromosome).

Being ordered, the first twenty chromosomes are selected, considered the "best" because of their achievement by reaching further distances. This selection method is called elitism. After being selected, these chromosomes are evaluated by the fitness function that checks to see if their distance is larger than a predetermined value. During the fitness assessment there is a count of chromosomes that are true to the situation. When this count reaches a value of 20, it is because it was found the most evolved population. While the count does not reach the maximum value, an initial population is recreated and then the whole process will begin again.

The state diagram of the Genetic Algorithm is shown in Figure 2.

![Figure 2 – State diagram (Genetic Algorithm)](image)

At the end of this algorithm we have a tripod that will have a movement uniform. This walk of tripod is being shown in Figure 3.
Figure 3 – Representation of the tripod intelligent agent’s movement

SOLUTION THROUGH GRASP

In GRASP we used the tripod creature as a solution and this will be made up of nine parameters.

In the construction phase, parameter values are initialized randomly creating a standard solution, starting a cycle. During this, the solution set is replicated, overwriting the values of each parameter (according to their range of values specified above in Table 1).

The first three parameters vary between 1 and -1, and the choice between them is made randomly. The following three parameters vary between -140 and 140, totaling a universe of 281 possible values. Each parameter of this was divided into four parts due to the large computational effort that would be required, making it feasible to work with this division. Finally, the last three parameters have a range from 0 to 77, creating a universe of 78 possibilities. For each parameter, data were collected for a shortlist. At the end of the cycle the top ten values of each parameter are selected on the shortlist, which were classified according to the distance that the solution come by starting thus a new cycle.

At the stage of local search, a new solution is generated where each parameter will be achieved through random values on the shortlist (collected in the previous phase) of its referent parameter. And finally, this generated solution is the final result of the first iteration. The stopping criterion is satisfied after three complete cycles of the algorithm.

The state diagram of the Genetic Algorithm is being shown in Figure 4.

![State diagram (GRASP)](image)

Figure 4 – State diagram (GRASP)

SIMULATIONS RESULTS

Several tests checking runtime were conducted, CPU and GPU processing, RAM consumption and quality of the solution found by the algorithm.

These tests were done on a computer with Intel (R) Core (TM) i5-4210U CPU @ 1.70GHz 1.70GHz, Installed memory (RAM) 8.00GB, Nvidia GeForce 820M and 64-bit Operating System, processor-based x64. The very Unity3D provides a Profiler Window that provides performance data of the algorithms during its execution. From it the following results were obtained:

- The GRASP had a latency three times greater than the Genetic Algorithm. While the GRASP spent an average of three minutes to compile and run the whole algorithm, GA spent an average time of 1 minute.

- According to Figure 5, Genetic Algorithm consumed more RAM than GRASP. In total it was used 137.6 MB memory, while the GRASP consumed 46.5 MB.

![Figure 5 – Comparison of memory usage when processing the Genetic Algorithm and GRASP algorithm](image)

The graphs of Figures 6 and 7 are to measure the amount of CPU and GPU processing that has been spent during the rendering, animation time expenditure, and algorithms’ logic. Being that the top of each image represents the genetic algorithm and the lower of GRASP. In addition, there is a look at garbage collection, and physical functions and sub-functions of each algorithm.

![Figure 6 – Comparison of CPU usage during processing of Genetic Algorithm and GRASP algorithm](image)

![Figure 7 – Comparison of GPU usage during processing of the Genetic Algorithm and GRASP algorithm](image)
The tripod agent of the Genetic Algorithm ran a final distance greater than the tripod of GRASP. The result of this the quality of the solution found by the algorithm, running ten times after the two algorithms is shown in Figure 8.

Figure 8 – Graph of the final distance traveled by the intelligent agent after the end of execution of each algorithm

The parameter vector (genes) that was returned at the end of execution of the genetic algorithm was used as input in a pseudo simulator of autonomous movements, also in the Unity3D engine and the result was satisfactory, while the same was done with the final tripod vector parameters of the GRASP algorithm and the result thereof was not satisfactory. This satisfaction is given when the tripod creature walks more naturally possible while maintaining some stability in its walk.

Evaluating the results (Table 2) it was possible to consider that the genetic algorithm is better regarding the performance of the CPU and GPU, runtime and quality of the solution, where the intelligent agent traveled a greater distance than the agent GRASP. However, GRASP consumed less memory than other algorithm.

<table>
<thead>
<tr>
<th>Test</th>
<th>Genetic Algorithm</th>
<th>GRASP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time of execution</td>
<td>1 minute</td>
<td>3 minutes</td>
</tr>
<tr>
<td>CPU Usage</td>
<td>16ms (60FPS)</td>
<td>33ms (30FPS)</td>
</tr>
<tr>
<td>GPU Usage</td>
<td>16ms (60FPS)</td>
<td>33ms (30FPS)</td>
</tr>
<tr>
<td>Memory</td>
<td>136.7 MB</td>
<td>46.5 MB</td>
</tr>
<tr>
<td>Final Distance</td>
<td>15 to 20</td>
<td>0 to 10</td>
</tr>
</tbody>
</table>

Figure 9 illustrates the evolution of the genetic algorithm. Each part of the figure represents a generation and we can observe the increase of the distance from the point of origin (highlighted by red line).

Figure 9 – Run of Genetic Algorithm

Figure 10 illustrates the performance of GRASP. Each part of the figure represents an algorithm stage and noticed that the distance traveled by the agent at the end of the last stage is much lower than the GA.

Figure 10 – Run of GRASP Algorithm

CONCLUSIONS AND PERSPECTIVES

The aim of this paper was to compare the genetic algorithm and the GRASP algorithm applied to the autonomous movement of an intelligent tripod agent.

According to the tests clearly it can be seen the best performance of GA compared to GRASP. Even with a larger universe of objects in the scene (in accordance with the tests done) it can stand out better in the tests in relation to GRASP.

Comparing the two algorithms, it is possible to realize that the Genetic is better than GRASP. To be initialized with random values, the GRASP can not find an optimal solution, while the Genetic Algorithm can. This is because the GRASP always seeks to find the best value for each parameter, without analyzing the tripod’s behavior completely (best place). In this case, this is not good, since when it improves a parameter, it ends up worsening the performance of another successively making the result not as satisfactory.

Meanwhile, the genetic algorithm seeks to find the best set of values for the genes by analyzing the tripod behavior as a whole, improving this solution until it is closer to a movement of a natural motion.

Future prospects include making the walk as natural as possible in other types of surfaces (regular and irregular) and propose a new model for the GRASP algorithm, since it did not obtain good results. Also, to evaluate the performance of these two algorithms in intelligent agents with other physical structures (biped and quadrupeds).

REFERENCES

Arakawa, Takemasa, and Toshio Fukuda. 1996 “Natural motion trajectory generation of biped locomotion robot using genetic


WEB REFERENCES

Boston Dynamics- http://www.bostondynamics.com/
Profiler Window (Unity 3D) - http://docs.unity3d.com/Manual/ProfilerWindow.html
Unity3D – http://unity3d.com/

BIOGRAPHIES

DÁMARIS DA SILVA BENTO was born in Carangola, Brazil and studied at the Instituto Federal Sudeste de Minas Gerais - Campus Rio Pomba, where he developed an extension project, working with electronic games in assistive education for disabled students. It is voluntary in scientific initiation project and the membrane Emcomp, Junior Company of the course of Computer Science since 2013, where he serves as vice president and front-end developer.

BARBARA RODRIGUES was born in Manhuaçu, Brazil Computer Science student at the Instituto Federal Sudeste de Minas Gerais - Campus Rio Pomba where is a fellow in Scientific Initiation by CNPq project and the membrane Emcomp, Junior Company of the course of Computer Science,since 2013, where he serves as Chairman of the Supervisory Board, front-end and back-end developer.

JULIANO ALVES was born in Cataguases, Minas Gerais, Brazil and studied at the Federal Institute of Education, Science and Technology at Southeast of Minas Gerais - Campus Rio Pomba, and he developed an initiation project named: "Creating a Groupware: A Case Study Evaluating the Main Possibilities ", He participates in Scientific initiation project and is member of the study and Free Software Development Group, IFGnu of computer science course, since 2012, where he serves as president and front-end and back-end developer.

BRUNO SOUSA is a Bachelor course student in Computer Science at the Federal Institute of Education, Science and Technology Southeast Minas Gerais - Campus Rio Pomba. He is a member of the research group and open source development IFGnu since 2012, and operates with research in software engineering, web services, SOA, front-end development and back-end development.

ANGELO NEVES is a Bachelor course student in Computer Science at the Federal Institute of Education, Science and Technology Southeast at of Minas Gerais - Campus Rio Pomba. He is a member of the research group and open source development IFGnu since 2012, and operates with research in software engineering, web services, android development, front-end development and back-end development.Also works at distance education center of this institute as a trainee.

LINDOMAR MATOS is a Computer Science student at the Instituto Federal in Minas Gerais. In 2013 the student
studied abroad at the University of Pisa - Italy, being the first student of the Campus Rio Pomba to benefit from the Science Without Borders Program. Recently he became the president of Emcomp (a Junior Enterprise on campus).

ALEX F. V. MACHADO has PhD in Computer Science at the Universidade Federal Fluminense. He is professor of Bachelor of Computer Science at IFSUDESTE-MG, and Coordinator of the Laboratory for Interactive Multimedia IFSUDESTE-MG, of the Programa de Educação Tutorial of Ministério da Educação (PET-MEC).
PREDICTING 2-WAY FOOTBALL RESULTS BY MEANS OF DATA MINING

João Gomes¹, Filipe Portela¹ ², Manuel Filipe Santos¹, António Abenha¹ and José Machado¹
¹Algoritmi Research Centre, University of Minho, Portugal
²ESEIG, Porto Politechnic, Porto, Portugal
joaogomes0991@gmail.com, {cfp,mfs}@dsi.uminho.pt, {abelha,jmac}@di.uminho.pt

KEYWORDS
Decision Support Systems, Data Mining, Football Games Prediction, Decision Support Systems for Football Betting, Knowledge Discovery in Database, 2-way result, Football Bets, eBusiness, Intelligent Systems

ABSTRACT

In the last decade, has been found an increase in the number of bookmakers, particularly in the online market (eBusiness). It is possible deducing that this activity is profitable for them and consequently damaging to their users. Nowadays, football is considered one of the most popular sports. Regarding the betting world it was acquired an outstanding position, which moves millions of euros during the period of a single football match. The lack of profitability of football betting website users has been stressed as a problem. In accordance with the stated arises here, an opportunity to explore. This lack gave origin to this research proposal, which is going to address the possibility of existing a way to support the users on their online bets, in order to improve their results and profitability. A football match could be analysed from the perspective of several types of statistical data, which do not have a direct influence on the final match result. This research work has the aim of helping to improve the performance of online football bets, by providing users statistical data that may be important to take into account, at the time of doing their own bets. In this work it was possible introduce data mining models which are able to predict 2-way results (home team win / draw or visitor team win) with 96.2% of sensitivity and a good level of accuracy (74.8%). These models are prepared to be the base of an Intelligent System.

INTRODUCTION

The use of Information System and Technologies (IST), which supports business activities is recognized as eBusiness. This type of business represents an exchange of products and services, between businesses, groups and individuals using electronic platforms. E-Business uses the internet as a vehicle to enable external activities and business relationships with individuals, groups and other businesses (e.g. selling products or provide services through the internet).

This paper addresses a particular market of e-business – the online betting markets. The Internet has opened a new market sector and gave birth to well-established online betting business, along with other e-business models that generated.

In the recent years, the emergence of online betting sector, lead to the conclusion that it is a very profitable business activity for these websites providers and may be unfavourable to its users. This new market increased has been gaining a relevant position in the e-business world. Online betting websites are often regarded as a profitable business for their owners. In order to increase the users’ profits, a new project arises. Firstly, this project includes a more theoretical chapter given by the literature review, explaining essential aspects for the development of the study followed by a practical approach. The practical part is focused in developing and present data mining models, in order to help the decision making process of a user in a particular bet, regarding a certain football game.

The project is focused in developing an intelligent system to e-business and studying the influences that statistical analysis of various events have in a game. Through the technological processes some patterns were identified knowing beforehand that there are several factors impossible to control, leading to suggest a correct bet. Briefly, the goal of this work is getting models with good predictive capabilities aiming to support the user in order to perform the best bet in a particular football. By making a comparison of this project, with the market available tools it is possible to observe a novelty, the use of data mining techniques to predict football online bets. This paper is focused in presenting the first results achieved. Using data mining models and the data collected from finished games (English Premier League 2000-2014) it was possible to induce models achieving 96% of sensitivity to predict a 2-way result - if the away team will win or not.

The paper is divided in seven sections. After a brief introduction, to the subject a background is provided based on the literature review and similar projects. Thirdly, it is an explanation of the research methodology. The forth section presents the Intelligent Decision Support System, which follows the methodology presented in the previous section. On the next topic, the main results are presented and discussed. Lastly, some conclusions are taken and future work presented.

BACKGROUND

Knowledge Discovery in Database

Knowledge Discovery in Database (KDD) is the process of identifying valid, novel, potentially useful and ultimately understandable patterns in data (U Fayyad, Piatetsky-Shapiro, & Smyth, 1996). It is an organize and automatic process, with an easy exploratory analysis and modelling that occurs in a given data repository (Maimon, Oded; Rokach, 2010). It is an interactive process, where the interaction is required at various stages, by the decision maker. It can contain loops between any two steps (U Fayyad et al., 1996). The basic framework is composed by nine steps (Um Fayyad, Piatetsky-Shapiro, & Smyth, 1996), which can be divided in five main steps: Selection,
Pre-processing, Transformation, Data Mining and Evaluation (Maimon, Oded; Rokach, 2010).

Data Mining

According Turban et. al (E Turban, Sharda, & Aronson, 2008) Data Mining will become so important that organizations cannot give up any information collected about their customers, risking to be out of business. Consider it, then, the next organizational strategic weapon. According J. Han et al. (Han, Kamber, & Pei, 2012), DM is the process of discovering patterns and interesting knowledge in large amounts of data.

Turban et al. (E Turban et al., 2008) initially defined data mining as a process through which patterns in data are found, but the definition was modified thus increasing its scope to a data analysis that has the objective to increase the efficiency and effectiveness of organizations. DM contains technical activities that can be subdivided into two major focuses of research, according to the analysis to be achieved, can be interpretive or predictive analysis (Vercellis, 2009).

This is a work that has the objective to make prediction in football game, so classification techniques were used. This task has the purpose of predicting the value that any random variable will assume, or else estimate the probability of a future event occur. Estimating therefore the value of a variable called 'dependent', 'target', 'response' (Tuffery, 2011). Depending on the nature of what it is intended to predict, the forecast can be divided into two more specific names (Efrain Turban, 2010), classification, if what you want to predict is a label of a class, for example, if a team will score [0-3] or more than 3 goals, and regression, if what you want to predict is a real number, for instance, the number of goals (e.g. 4).

Decision Support Systems

Decision Support Systems (DSS) are like an interactive computer system that supports managers to make decisions related attributes, goals and objectives, to solve semi-structured and unstructured problems (Nemati, Steiger, Iyer, & Herschel, 2002). They are, therefore, systems that support managers in decision making by providing and analysing several alternatives, analysing historical decisions and what influence these have on your organizational context giving support to problem solving (Shim et al., 2002).

The DSS follow as the basis for their development stages of the decision-making process. Simon (H. A. Simon, 1960) is the author of the methodology that brings together a greater consensus among the community. Initially he defines the decision-making process as having only three phases, the Intelligence, the Design and Choice. Years later, Simon (H. a. Simon, 1977) and many other authors defined a fourth phase Implementation, therefore, argued that the implementation of what had been previously decided was important enough to the point of creating an individual stage for the same.

The results achieved by the Data Mining models will be included in a Intelligent Decision Support System with pervasive features (Filipe Portela, Jorge Aguiar, Manuel Filipe Santos, Álvaro Silva, & Fernado Rua, 2013; Henricksen, Indulska, & Rakotonirainy, 2001; Portela, Santos, Machado, Abella, & Silva, 2013; Portela, Santos, & Vilas-Boas, 2012; Satyanarayanan, 2002). The processing and inducing tasks will be performed automatically and in real-time by intelligent agents, following a concept already implemented in the healthcare (Cardoso et al., 2014).

Football Gambling Systems

The gambling industry is a large industry, with many companies operating on different frameworks such as, the traditional betting companies, online companies and betting exchanges. As an example, is Betfair, Bwin and bet365 (respectively). Each online betting company has their own odds and betting exchanges. Sports, in particular football is one of the most common bets market presenting several betting possibilities (2-way and 3-way results; number of corners; number of goals; faults, among others).

After a research on forecasting systems for football games, a set of platforms aiming to support the decision, in terms of football matches results were found. The Intelligent Decision Support System (IDSS), intended to be developed in this project has a particular feature that was not found in any of these other platforms. The IDSS proposed is based not only in mathematical calculations. This system distinguishes from the other existing platforms, due the use of DM techniques.

The following platforms are websites containing the probability of occurring one of the three results (victory of the home team, draw or away team win), and other statistics that are also important. These systems suggest the bet which they consider ideal: http://spotwin.net/football-betting-system-7.html, http://soccervista.com/, http://www.predictz.com/, http://www.windrawwin.com, https://www.statarea.com/ http://vitibet.com/, http://pt.zulubet.com/, http://www.footwin.net/, http://www.forebet.com/. At the moment there is also some mobile platforms using only mathematical calculations: KickOff – Smart Betting Made Simple, Smart BET Prediction and FootWin – Sports Prediction. In general the existing systems do not have associated a confidence level or an accuracy of the results predicted.

In consonance, one of the goals of this project is increasing the gains by changing the way of how the calculations are done (using DM as base) and add a confidence indicator.

METHODOLOGIES, MATERIAL AND METHODS

This is a research project and it is following the Design Science Research (DSR) Methodology. DSR should lead to the production of a viable artefact in the form of a building, a model, a method or an instantiation. The goal is to develop a technology-based solutions to important and relevant business problems (Hevner, March, Park, & Ram, 2004).

To complement this methodology, a combination of two other methodologies was used: Cross Industry Standard Process for Data Mining (CRISP-DM) and the Simon phases of decision making. For example, the first phase is composed by a combination of the Intelligence phase of decision-making and Business Understanding of CRISP-DM methodology.

In terms of technology it was used KNIME and Weka.
INTELLIGENT DECISION SUPPORT SYSTEM

As it was previously mentioned this project was developed recurring to a combination of CRISP-DM methodology and decision-making phases. At this point is presented the work developed in each phase, by the combination of both methodologies.

Phase 1

This phase was a combination of the Intelligence phase of decision-making and Business Understanding of CRISP-DM methodology. Here it was identified the problem or opportunity, and an analysis was made to the environment by collecting information of what it is happening at the moment and after, with the purpose of making a comparison with what it is expected.

The problem which was identified is the lack of profit by users of long-term in sports betting. A DSS in this area should support the users to determine which bet is the most probable to happen. This problem is a semi-structured decision problem because there are data that the user will need to enter beyond the existing statistics data already presented in the dataset.

Phase 2

In this second phase a merge of the four other phases was made, one of decision-making, namely Design and three of CRISP-DM, Data understanding, Data Transformation and Modelling. This phase has the objective to help the decision maker setting his own objectives and analyses the risk and possible restrictions, defining finally the decision. In this phase it was decided to develop an intelligent system, using data mining models, which would be complemented with human perception, making a mixture of man/machine intelligence, in order to take the most reliable decision. On this stage, a detailed analysis of the data collected was made. This dataset was based in statistical data from fourteen seasons of the English Premier League, since 2000/2001 until 2013/2014 and each one had a spreadsheet file, every season has statistical information of each game performed.

The variables contained in that dataset are presented in Table 1. This dataset contains information from 5320 games.

<table>
<thead>
<tr>
<th>Original Variables</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>DATE</td>
<td>Match Date (Dd/Mm/Yy)</td>
</tr>
<tr>
<td>HOMETEAM</td>
<td>Home Team</td>
</tr>
<tr>
<td>AWAYTEAM</td>
<td>Away Team</td>
</tr>
<tr>
<td>FTHG</td>
<td>Full Time Home Team Goals</td>
</tr>
<tr>
<td>FTAG</td>
<td>Full Time Away Team Goals</td>
</tr>
<tr>
<td>FTR</td>
<td>Full Time Result (H=Home Win, D=Draw, A=Away Win)</td>
</tr>
<tr>
<td>HTHG</td>
<td>Half Time Home Team Goals</td>
</tr>
<tr>
<td>HTAG</td>
<td>Half Time Away Team Goals</td>
</tr>
<tr>
<td>HTR</td>
<td>Half Time Result (H=Home Win, D=Draw, A=Away Win)</td>
</tr>
<tr>
<td>ATTENDANCE</td>
<td>Crowd Attendance</td>
</tr>
<tr>
<td>REFEREE</td>
<td>Name Of Match Referee</td>
</tr>
<tr>
<td>HS</td>
<td>Home Team Shots</td>
</tr>
</tbody>
</table>

Table 1: Original Variables

After collecting the data a treatment and data processing phase was executed. For it, it was used an Extract Transforming and Loading (ETL) process which it is presented in the Error! Reference source not found..

![ETL Process](image)

**Figure 1: ETL Process**

This is a simple process, begins with the creation of two database tables, a temporary table and other one that would be filled later. The original dataset was uploaded to the temporary table. Then it was updated the fields “Date” and “FTR”. “Date” field was separated into three other fields, day, month and year. FTR is the final time result (1, X, 2).

The data presented in the dataset is mostly data that can only be obtained at the end of each game. Additionally, it was necessary to identify other variables that could be known before the football game begins. Due to that fact, several indicators were created for the home team and the away team. These indicators are averages of the variables that were in the original dataset. Finally, and before loading the final table, all existing null fields were removed and “FTR” field was transformed. “FTR” was transformed in only two results possibilities (2-way result) - '0' if the result is the home team win or draw and '1' if the result is the away team win. This transformation was made with the goal of increasing the winning chances probability and after observing the difficulty to predict the 3-way results (low accuracy results). Then it was charged to another table with the data from the temporary database. Some DM models were induced using statistic variables created:

- AVGGoals and AVGGoalsConceded;
- AVGShots and AVGShotsConceded;
- AVGShotsTarget and AVGShotsTargetConceded;
- LastFiveResults;
- LastFiveConfrontationResults.

In the next step classification DM models were induced with the data obtained after been processed and transformed. All the variables were considered in all the models. In this phase
ther is not any pre-defined scenario (DM engine automatically chosen the most significant variables to each model).

The DM models were induced using three different DM techniques, Naive Bayes, Decision Trees and Support Vector Machine and the algorithms NB, J48 and LibSVM respectively. In order to perform the test phase have been selected two distinct sampling methods, 10-folds Cross Validation (CV) and the percentage split (% Split) in which 66% of data were used to training the model and the remaining were used to test.

Phase 3

This phase is a combination between CRISP-DM Evaluation phase and the Choice phase from decision-making. At this phase the evaluation of the data mining models was made in order to choose which is the most suitable model.

To evaluate all induced models the confusion matrix was induced. This matrix is composed by the true positives (TP), which correspond to the number of examples positives thus classified, false positives (FP), which correspond to the number of examples positives classified as negatives, false negatives (FN), which correspond to the number of examples negatives classified as positives and true negatives (TN), which correspond to the number of examples negatives thus classified. Using the confusion matrix it was possible calculated several metrics. The most common metrics used in classification data mining problems are sensitivity, specificity, accuracy, precision and area under curve (AUC).

To evaluate the results obtained by the DM models, the five above measures were considered. As already mentioned, three different techniques were used to create the models: Support Vector Machines (SVM), Naive Bayes and Decision Trees (DT). For the first technique it was applied the LibSVM algorithm of Weka tool, for the second it was used the NB algorithm and for the third it was used the J48 algorithm.

To accomplish this task, two sampling methods were used: the percentage split (66.6 % of the data for training and 33.4% for testing) and 10 folds Cross Validation. Table 2 contains the obtained results by measures and each data mining technique used.

Table 2: Models Analysed

<table>
<thead>
<tr>
<th>Model</th>
<th>1</th>
<th>2</th>
<th>3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling Method</td>
<td>CV</td>
<td>% Split</td>
<td>CV</td>
</tr>
<tr>
<td>Algorithm</td>
<td>LibSVM</td>
<td>LibSVM</td>
<td>J48</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>95.4%</td>
<td>96.2%</td>
<td>91.6%</td>
</tr>
<tr>
<td>Specificity</td>
<td>22.1%</td>
<td>19.5%</td>
<td>27.7%</td>
</tr>
<tr>
<td>Accuracy</td>
<td>75.3%</td>
<td>74.8%</td>
<td>74.0%</td>
</tr>
<tr>
<td>Precision</td>
<td>76.4%</td>
<td>75.6%</td>
<td>77.0%</td>
</tr>
<tr>
<td>AUC</td>
<td>58.7%</td>
<td>57.9%</td>
<td>64.0%</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Model</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sampling Method</td>
<td>% Split</td>
<td>CV</td>
<td>% Split</td>
</tr>
<tr>
<td>Algorithm</td>
<td>J48</td>
<td>NB</td>
<td>NB</td>
</tr>
<tr>
<td>Sensitivity</td>
<td>95.8%</td>
<td>72.0%</td>
<td>72.2%</td>
</tr>
<tr>
<td>Specificity</td>
<td>20.5%</td>
<td>62.3%</td>
<td>62.0%</td>
</tr>
<tr>
<td>Accuracy</td>
<td>74.8%</td>
<td>69.3%</td>
<td>69.4%</td>
</tr>
<tr>
<td>Precision</td>
<td>75.7%</td>
<td>83.4%</td>
<td>83.1%</td>
</tr>
<tr>
<td>AUC</td>
<td>63.4%</td>
<td>74.0%</td>
<td>74.3%</td>
</tr>
</tbody>
</table>

After getting the results an assessment of the models was done. Firstly, it was possible to identify that does not exist a model that can be considered ideal, however they have different characteristics that may be considered interesting.

There is a value in one of the measures that stands out from the others, the sensitivity - with it is possible to say that when the model suggests the occurrence of the result '0' - reaches a considerable percentage of hit. This measure got a higher value in model 2, 96.2%. This is a very reasonable value, but on the other hand the specificity obtained values considered as lower than expected. Analysing these results it is possible to note that this is a model able to predict only one class (0).

This work represents the first phase of the project. The model chosen by now has sensitivity values that have to be taken into account, but in future the goal is to obtain more balanced values. The most accurate model predict both classes is model 1 with an accuracy of 75.3%. However, there are two models that did not obtained very good results, in other words, they are the most equilibrated, present the best precision and AUC values: model 5 and 6.

**DISCUSSION**

Through the analysis of the models it was possible to identify that the sampling method did not have a big influence on the results. The opposite occurred with the use of different DM techniques that resulted in the development of models with different values.

Each model had interesting values, Model 1 is the model with greater degree of accuracy (75.3%), the second model stands out from the others because it has a higher sensitivity (96.2%) and greater accuracy (74.8%), model 4 equals model 2 in the level of accuracy, the model 5 has a higher accuracy level (83.4%) and also the highest specificity (62.3%) and finally the model 6 is getting greater AUC (74.3%).

The choice of the best model was focused on two of the six models, the model 2 and model 5. Despite of the model 5 obtain more balanced values, it does not reach high values as model 2. Being the project goal helping bookmakers users to run less risk in order to get hit as many bets as possible, leading them to make a profit with this activity, the model 2 was then chosen.

The models with high sensitivity and low specificity normally have good performance when the positive classe is predicted. In this case, model 2 presents the highest sensitivities rates being able to hit 96.2% of the cases where the final result is '0' (home team win or draw). However this model is very bad to predict classe '1' presenting a specificity of 19.5%. In accordance with the sentence above mentioned it is only possible to take advantage of half of this model. The user should only bet on the result '0' (home team win or draw) when the model suggests, forgetting the second possibility of the class, '1' (away team win). So when the model suggests '1' is better to stay without betting. To avoid the high number of false negatives another model should be chosen (presenting a better accuracy and precision).

In sum, being this an IDSS all the possibilities can be given to the user, being him responsible to make his own choices. The system will present the prediction and the confidence level for each results. The bet suggestion can be
ranked according the profit, confidence or other metric defined.

This new possibility of introducing ranking metrics brings new bet opportunities to the users. They will have more information before making a bet. They will know more about the bet, due to the information provided by the IDSS and the reason underlying (variables associated which allow to conclude which the best bet is).

CONCLUSION

Due to the development of this project it was possible to verify that the use of data mining models can lead to successful results in football games bets. The objective of this project is to reduce the rate of wrong bets by the gambler, leading him to get bigger profits and running lower risks. The achieved results in this research gave good indications in order to continue a further research.

In this first phase the models obtained so far did not fulfilled all the expected results (the result of accuracy and precision metrics are lower). However, it was possible to obtain very interesting values, specifically in the case of sensitivity (predict "0") for which it obtained the value of 96.2% which is very satisfactory and motivating for a future research proposal. The number of false positives is the biggest concern at the moment, reaching values about 62.3%, which is well upper the expectations. This values are higher due to the difficult to predict the classe "1".

In fact, these models are very good to predict the probability of the home team victory or a draw result, but not so good to predict the away team victory. Briefly, a gambler should follow the system indication when the bet is "0".

Lastly, if we use these models at the betting time, by keeping it in pace with possible online bets it is possible to make probabilities and bets in real-time, before the game starts.

FUTURE WORK

Future work will begin with the development of a prototype of an intelligent decision support system and a further integration of the DM models.

After developing them, the goal is to develop new models and test in order to achieve the better predictive results.

Concluded the test phase it is intended to: Improve DM models, testing different techniques, introducing new variables, for example, the rest time between each game and trying to determine if there is a way to improve the prediction values of the result. Study the financial impact of the system; Make experiences about how much a gambler can win by using this system.

ACKNOWLEDGEMENT

This work has been supported by FCT - Fundação para a Ciência e Tecnologia within the Project Scope UID/CEC/00319/2013

REFERENCES


Maimon, Oded; Rokach, L. (2010). Data Mining and Knowledge Discovery Handbook (2nd ed.).


AUTHOR BIOGRAPHIES

João Gomes is a MSc Student of the Algoritmi Research Center, University of Minho, Braga and is developing his research work in the topic "Data Mining Models to predict football results".

Filipe Portela holds a PhD in Information Systems and Technologies. He belongs to the R&D Centre ALGORITMI and is developing his research work in the topic “Pervasive Intelligent Decision Support in the Critical Health Care”. Actually he is an Asstent Professor (invited) of the Department of Information Systems, School of Engineering, University of Minho, Portugal and a Guest Lecture in ESEIG.

Manuel Filipe Santos holds a PhD in Computer Science – Artificial Intelligence. Actually he is an Associate Professor in the Department of Information Systems, School of Engineering, University of Minho, Portugal. He is the head of the Intelligent Data Systems group in the R&D Centre Algoritmi.
José Machado is an Associate Professor with Habilitation of the Department of Informatics of the University of Minho, in Braga, Portugal. He got his PhD in Informatics, in 2002, and is now the Director of the Computer Science and Technology R&D Center (in Portuguese CCTC).

António Abella is an Assistant Professor at University of Minho, Department of Informatics, in Braga, Portugal and he got his PhD in Informatics, in 2004, and is a researcher of Algoritmi Research Center. He is the header of the Electronic Health Records Research Group.
PETRI NET SIMULATION
QUANTITATIVE ANALYSIS OF THE PUBLISH/SUBSCRIBE PARADIGM IN THE CONTEXT OF WEB SERVICE RESOURCES WITH TIMED COLORED PETRI NETS

Valentín Valero
Hermenegilda Macià
Gregorio Díaz
School of Computer Science
University of Castilla-La Mancha
Albacete, 02071 - SPAIN
email: {Valentin.Valero,Hermenegilda.Macià,Gregorio.Diaz}@uclm.es

KEYWORDS
Quantitative analysis, Timed Colored Petri Nets, Web Services, Web Service resources

ABSTRACT
In this paper we present a formal model based on Prioritized-Timed Colored Petri Nets capturing the different operations offered by the Publish/Subscribe (P/S) paradigm in the context of distributed resources, which include the capability for publishing and discovery of resources, as well as for manipulating and subscribing to these resources. CPN Tools are then used to accomplish a quantitative analysis of this model, so we can predict the system behavior and discover potential problems in P/S systems before the implementation.

INTRODUCTION
The publish/subscribe paradigm provides a loosely coupled form of interaction in large scale settings, where a set of (distributed) resources are made available (published) by some providers, and some subscribers can then register their interest in a topic or a pattern of events, receiving notification messages when these events occur. Eugster et al. (Eugster et al., 2003) and also Lin and Plade (Lin and Plade, 2003) have made a taxonomy of Publish/Subscribe systems, with a comparison between the different alternatives. There are basically two categories, the subject-based vision, and the content-based systems. In the subject-based systems the clients join to groups of interest, and all of them are notified of the events related to that group. In content-based systems, in contrast, a subscriber indicates a query or predicate related to the resource contents, and a notification is then sent to the subscriber when this predicate becomes true. In this paper we consider a content based approach. Therefore, subscriptions will have a predicate associated related to the WS-resource property values. Thus, notifications will be immediately sent when the resource state makes true this predicate. Some formalizations have been made of the Publish/Subscribe paradigm, but most of them focus on subject-based systems. (Baldoni et al., 2003) have defined a formalization based on the following process operations: publish, notify, subscribe and unsubscribe. They establish several information availability models, proving completeness and minimality for the computations they produce. They consider the subject-based approach, so no predicates are considered for notifications, which occur when items are published. The subject-based approach is also considered by (Zanolin et al., 2003). In this case, the application-specific components are modeled as UML statechart diagrams and the middleware in charge of publication and notification of events is supplied as a configurable predefined component. The SPIN model-checker is then used to verify the properties of interest. This work was later extended by (Baresi et al., 2007), by including some additional features such as message reliability, message ordering, message priorities, etc. (Garlan et al., 2003) have also applied model-checking techniques for the analysis of a generic Publish/Subscribe framework. They have built a tool that works on a parameterized state machine model, which accepts as input a set of component descriptions together with a set of properties, thus producing a model that can be checked with the Cadence SMV model checker.

More related to our work, (Abidi et al., 2011) have developed a CPN model for the Publish/Subscribe paradigm, in this case in the context of a specific Grid protocol (Bonjour/GRID middleware), which supports resource discovery and coordination in a desktop Grid computing environment. However, they do not use Web Services, Web Service Publishing or Web Service Discovery standards in their work, since it is focused on a specific application.

Our work in this paper, in contrast with these works, is focused on providing a rigorous model using Prioritized-Time Colored Petri Nets capturing the main elements for the publishing, discovery and management of WS-
resources on the basis of Web Services standards, such as UDDI (Universal Description Discovery and Integration), WSRF (Web Services Resource Framework) and WSN (Web Services Notifications). There is another standard for Web services notifications from the W3C, namely WS-Eventing, which has many similarities with WSN, so any of them could be considered as reference for the subscribe/notifify operations.

Another Petri net representation of the subject-based Publish/Subscribe systems was made by (P. Hens and Backer, 2011). In this case, time restrictions are not included in the model, and thus an ordinary Petri net model establishes the connection between the publishers and subscribers, in order to send the corresponding notifications when the events are published.

We can find other works in the literature that use formal techniques for performance analysis, but focused on different application domains. For instance, Stochastic Well-Formed Nets (SWNs) are applied in (Bernardi and Meregher, 2007) in the context to the UML Profile for Schedulability, Performance and Time (UML-SPT), with the goal to provide software engineers with a method for the computation of performance metrics, such as response time, sojourn time and throughput on UML design.

The issue of adding compositionality to a class of Stochastic Colored Nets has been studied in (Ballarin et al., 2000), where the authors use Parametric Stochastic Well-Formed Nets (PSWNs) and compositional modeling in a general framework to model hardware and software components with the purpose to increase modularity and reuse of the modelling efforts. Resources are here considered, but not any mechanism for either their publication, discovery or subscription is considered.

From all these works it becomes obvious that the way in which the publish/subscribe systems are modeled varies considerably depending on the specific model goals. In this paper, we intend to use the CPN Tools capabilities for the analysis and simulations of timed systems modeled by using Prioritized-Timed Colored Petri Nets. We use a discrete time model because we have needed a discrete direct time control in order to be able to reevaluate the guards that condition the firing of transitions. This would not be necessary if CPN Tools would update the guards evaluations when the global clock is increased, and not only when some transitions fire. Some of the involved operations have delays that follow a negative exponential distribution, so we introduce a discretization of the times generated by these distributions.

A mechanism to create and publish distributed resources in (discretized) exponential times is then introduced, where the resources are identified by a textual name. The participants or clients follow roles that can also be generated by (discretized) exponential times. These participants will discover the resources by using their textual names.

Discovery of resources provide the clients with the required EPRs (End Point References) to access and manipulate these resources. They can read or modify the resource property values, as well as the resource expiration time (lifetime). These operations are accomplished at random times, also selected by using a negative exponential distribution.

The same occurs for subscriptions, in which the clients indicate a predicate that depends on the resource property values, with the purpose of being notified when the predicate becomes true. These subscriptions have also a (deterministic) lifetime associated, which means that once this time has elapsed, the corresponding subscription is canceled if it has not been notified before. A work modeling a previous version of this Publish/Subscribe paradigm in the context of distributed resources, without Markovian delays in the generation of both resources and roles can be found in (Valero et al., 2015). In that work the core of the PTCPN model was validated by using CPN Tools.

The rest of the paper is organized as follows. The Publish/Subscribe CPN-model is then introduced by parts, after which the performance analysis of the complete model by using a specific case study is presented. The paper finishes giving some conclusions and the possible lines of future work.

BACKGROUND

In this section we establish the required background for both the WS-resource contents-based Publish/Subscribe paradigm and the prioritized-timed colored Petri net formalism that we use.

WSRF and the Publish/Subscribe Paradigm

The Web Services Resource Framework (WSRF) is an OASIS standard (Organization for the Advancement of Structured Information Standards), which defines a framework for modeling and accessing persistent resources using Web services. This approach consists of a set of specifications that define the representation of WS-resources manipulated by Web services. WS-resources are described by the so-called Resource Properties Documents, which are XML specifications that contain all the relevant resource information. This document is a projection of the actual state of the WS-resource and serves to define the structure upon which query and update messages are directed. Thus, any operation that manipulates a resource property via the WS-resource properties document must be reflected in the actual implementation of the WS-resource's state. The WSRF standard provides us with operations to read or modify the resource properties (getProp and setProp, respectively), as well as to obtain or modify the resource lifetime (getTime and setTime, respectively), but no indication is made about the way in which resources are created and made visible. It is assumed that WS-
resources are created by some external mechanism or through the use of a WS-resource factory, which creates the resource and establishes an association with a Web service, returning an endpoint reference (EPR), which can thereafter be used to direct requests to the WS-resource.

We enrich the model with the publish/discovery registry-based mechanism. These operations, Publish and Discover resemble the save_service and find_service operations of UDDI. We therefore provide a Publish operation to publish a WS-resource, indicating its EPR, tag (textual resource type identifier), initial value and initial lifetime. Notice that there can be several distinct implementations of a WS-resource (e.g., a printing service may be offered using different printers), so the discovery mechanism will only return the EPR of one of them. Thus, a Discover operation is also provided, which allows us to obtain a WS-resource from the Registry, according to a given tag.

Resources can be destroyed either by invoking the operation Destroy or because their lifetime has expired. The Destroy operation is equivalent to reassigning the resource lifetime to zero, so we can use this operation as a way to destroy resources.

WSRF can be complemented with WSN (Web Services Notifications) (Niblett and Graham, 2005), which defines a set of specifications to standardize the Subscription/Notification mechanism, with the purpose of allowing clients to subscribe to WS-resources and be notified about specific changes in the resource state. A Subscribe operation is therefore provided, in which the client indicates the EPR of the resource and the TopicExpression that indicates the condition upon which the notification must be sent. In addition, subscriptions may have a finite duration, after which the subscription is canceled. There are some other features of WSRF that will not be considered in this paper, such as the insertion and deletion of properties for existing WS-resources, the aggregation of multiple WS-resources or Web services into ServiceGroups, or the fault handling mechanisms.

Prioritized-Timed Colored Petri Nets

We use prioritized-timed colored Petri nets, which are a prioritized-timed extension of colored Petri nets (Jensen, 1997), the well-known model supported by CPN Tools, developed by the CPN group at the University of Aarhus. The CPN modelling language can be used to investigate both functional and performance properties, in contrast to other modelling tools that can only be used to analyse either functional or performance properties.

In this model, places have an associated color set (data types). Each token has an attached data value (token color), which belongs to the color to which the token is associated. Furthermore, tokens have a timestamp associated, indicating the time at which they will be available in order to fire some transitions.

A discrete global clock represents the total time elapsed in the system model. Arrows can have inscriptions (arc expressions), constructed using variables, constants, operators and functions. To evaluate an arc expression we need to bind the variables, which consists of assigning a value to the variables that appear in the arc inscription. These values are then used to select the token colors that must be removed or added when firing the corresponding transition.

Arc expressions can also have associated time information both for place-transition and transition-place arcs, but normally they are only used in transition-place arcs to indicate a time that must be added to the current global clock value in order to get the timestamp of the tokens produced by these arcs. Actually, when all the output arcs of a transition have the same time inscription, there is a shorthand notation in CPN Tools by which this time information is associated with the transition instead of the output arcs.

Transitions can also have associated guards, which are Boolean expressions that can prevent their firing. Thus, when a transition has a guard, it must evaluate to true for the binding to be enabled, otherwise the binding is disabled and the transition cannot fire.

Definition (Prioritized-Timed Colored Petri Nets)

A prioritized-timed colored Petri net (PTCPN) is a tuple \((P, T, A, V, G, E, D, \pi)\), where

- \(P\) is a finite set of places, with colors in a set \(\Sigma\). Remember that tokens have associated a non-negative integer value indicating their timestamp.

- \(T\) is a finite set of transitions \((P \cap T = \emptyset)\).

- \(A \subseteq (P \times T) \cup (T \times P)\) is a set of directed arcs.

- \(V\) is a finite set of typed variables in \(\Sigma\), i.e. \(Type(v) \in \Sigma\), for all \(v \in V\).

- \(G : T \rightarrow BOOL.EXPR_V\) is the guard function, which assigns a Boolean expression to each transition, i.e. \(Type(G(t)) = Bool\).

- \(E : A \rightarrow EXPR_V\) is the arc expression function, which assigns an expression to each arc, constructed using variables, constants, operators and functions.

- \(D : T \times P \rightarrow NAT.EXPR_V\), which is the token delay function, which associates a discrete time delay expression to each tp-arc. This delay is applied to all tokens produced by that arc. An omitted time delay is a shorthand for a zero time-delay.

- \(\pi : T \rightarrow \mathbb{N}\) is the priority function, which assigns a priority level to each transition. We specifically use four priority levels, from \(P1\) to \(P4\), \(P1\) being the highest priority and \(P4\) the lowest one.
We use the classical notation on Petri nets to denote the precondition and postcondition of both places and transitions:

$$\forall x \in P \cup T : \quad \text{pre}(S) = \{ y \mid (y,x) \in A \} \quad \text{post}(S) = \{ y \mid (x,y) \in A \}$$

**Definition (Markings)**

Given a PTCPN $N = (P,T,A,V,G,E,D,\pi)$, a marking $M$ is defined as a function $M : P \to B(\Sigma)$, which assigns a multisets of colors to each place (which can be empty).

A timed marking of a PTCPN $N$ is a pair $(M,x)$, where $M$ is a marking of $N$ and $x$ is the current system time instant. A marked prioritized-timed colored Petri net (MPTCPN) is then defined as a triple $(N,M,x)$, where $N$ is a PTCPN, and $(M,x)$ a timed marking of it.

We omit the formal definitions of both the bindings and the firing rule, which can be found in (Jensen and Kristensen, 2009). Informally, a transition binding is just a function that assigns values to the variables that appear in a transition or in the arcs connected with it. Given a timed marking $(M,x)$ and a transition $t$, we say that a transition binding of $t$ is enabled at an instant $x' \geq x$ when $G(t)$ is true under this binding, for every $p \in \bullet t$ we have enough tokens in $M(p)$ with colors matching with the indicated in $E(p,t)$, no other transition $t'$ has a binding fulfilling these conditions with $\pi(t') < \pi(t)$ and $x'$ is the smallest time for which there exists a binding for $t$ fulfilling these conditions. The firing of an enabled transition binding is non-deterministic, and it is always instantaneous, i.e. takes no time. When a time delay is associated to a transition, this can be interpreted as the duration of the operation modelled by this transition. The generated tokens from the firing of a transition with a delay $x$ will not be available until $x$ time units later. Notice that with the firing of transition $t$ we remove the selected tokens matching with $E(p,t)$, and the new tokens on the output places has colors according to the expressions $E(t,p')$.

![PTCPN Diagram](image)

**Example 1**

The PTCPN depicted in Figure 1 describes a system in which a bounded number of clients are randomly generated which request services of three different types (A, B or C). Two servers are then used to attend the client’s requests, but Server1 can only serve requests of types A or B, whereas Server2 can attend all types of requests. Place Capacity is used to produce a finite number of client’s requests, so we use one untimed integer-colored token (INT) in this place, whose value indicates us the remaining number clients that can yet be produced. Thus, each firing of transition Request decrements this counter by one unit, and the guard $k > 0$ will stop the generation of client’s requests when the counter on the place Capacity is zero. Place Clients contains three untimed string-colored tokens, with values “A”, “B” and “C”, and each firing of transition Request can pick whichever to produce a request of one of these types. Place Waiting contains the produced requests, whose timed tokens are colored with the strings identifying the request types. We use the function expTime defined in Table 1 in order to generate discretized delays obtained from a negative exponential distribution of parameter $r1$.

Requests on the place Waiting are then served either by Server1 (only types A, B) or Server2, but Server1 has a priority over Server2. We use again negative exponential distributions for both service times, with parameters $s1,s2$, respectively. Finally, both places Busy1 and Busy2 contain one timed-empty colored token (without information) to enforce the sequential firing of both transitions and thus both servers can only attend a request at a time.

<table>
<thead>
<tr>
<th>Table 1: Declarations of Example 1.</th>
</tr>
</thead>
<tbody>
<tr>
<td>colset INT = int;</td>
</tr>
<tr>
<td>colset UNITT = unit timed;</td>
</tr>
<tr>
<td>colset Rec = string timed;</td>
</tr>
<tr>
<td>val PR=&quot;1&quot; &quot;A&quot;=&quot;1&quot; &quot;B&quot;=&quot;1&quot; &quot;C&quot;;</td>
</tr>
<tr>
<td>fun expTime (mean : int) =</td>
</tr>
<tr>
<td>let</td>
</tr>
<tr>
<td>val realMean = Real.fromInt mean</td>
</tr>
<tr>
<td>val rv = exponential((1.0/realMean))</td>
</tr>
<tr>
<td>in</td>
</tr>
<tr>
<td>floor (rv+0.5)</td>
</tr>
<tr>
<td>end;</td>
</tr>
</tbody>
</table>

**PTCPN MODELING OF PUBLISH/Subscribe**

In this section we describe the PTCPN model for the WS-resources and the Publish/Subscribe mechanism. We present the model by parts. Thus, we first present the PTCPN for the automatic generation of both resources and participants, then the publishment and basic management of WS-resources, after which the discovery and subscription mechanisms are modeled, and finally, notifications are included in the model. The per-
formance analysis of the complete model is accomplished in the next section, by using the CPN Tools capabilities for statistical analysis.

Generation of Resources and Participants

The PTCPN for the generation of both resources and participants is shown in Figure 2. Transition GenResource produces the resources that feed the generic model, using the information provided by the initial markings in the places ResEPRs, ResValues and ResLifeTime, which respectively will provide us with the resource EPR, initial value and lifetime. In order to have a bounded model, we have included the place MaxRes whose initial marking restricts the maximum number of generated resources. A similar approach is used to generate the participants that will feed the model, by the transition GenParticipant. Places RoleID, RoleCond1, RoleCond2 and RoleSubsTime are here used to provide the participant identifier, the subscription condition in the form of an interval of values \([v1, v2]\) and the subscription duration. Once again, place MaxParticipants is used to limit the number of generated participants.

Notice that both transitions GenResource and GenParticipant have a negative exponential distribution associated in order to generate the resources and participants with a Markovian model.

WS-resource Modeling

The PTCPN that models the WS-resource management is shown in Figure 3. From Figure 2 if follows that place Resources (left-hand side of Figure) contains the colored tokens representing the resources in the system, which will eventually be published. The information attached to each one of these tokens is a tuple \((EPR, R, v, lt)\), where EPR is the End Point Reference of the WS-resource \(^1\), \(R\) is a textual tag identifying the resource type, \(v\) is the initial value, and \(lt\) its initial lifetime. Resources are then published by firing the transition Publish_ok, but notice that in the event that there exists an already published resource in the Registry with the same EPR, the transition Publish_fail will instead fire (it has greater priority), indicating a failure in the resource publication. We use the CPN Tools function intTime(), which provides us with the current clock value, so the tokens produced in the Registry place will only be available until the time indicated in its last component: \(intTime() + t\). Discrete time elapsing is modeled by transition TimeStep, which updates the value of the token on the place TimeControl, which is initially marked with one token with value 0. This mechanism of direct control of time elapsing is needed due to a problem with guard transition evaluation, since they are not reevaluated as time elapses in CPN Tools unless a transition fires. Transition ResourceExpire will then fire once the value of the token on TimeControl exceeds the time indicated on the last component for some token in the Registry, i.e., resources are unpublished at the next time instant, after its lifetime has expired. The firing of ResourceExpire is enforced by its priority, and the corresponding resource token is moved to the place ExpiredResources. In order to avoid finite behaviours, control guard has been included in the transition TimeStep, which restricts the maximum model time.

The remaining transitions model the basic operations on WS-resources: SetProp, GetProp, SetTime and getTime. We use in this generic model a function \(nV()\)

\(^1\) We use non-negative integer numbers as EPRs, and also as resource values.
Discovery and Subscription Modeling

The WS-resource discovery and subscription part is modeled as indicated in Figure 4. The place Participants contains (Figure 2) the tokens representing the participants and the resources they intend to use. These tokens have as colors tuples indicating the participant id, a resource tag, two integer values defining an interval and a subscription lifetime. They have as timestamp the time instant at which the corresponding participant intends to perform the discovery and subsequently the subscription at the obtained WS-resource.

The transition Discover obtains one of the tokens in the Registry that matches with the indicated tag, producing a token in the place SubscriptionRequests, labeled with the corresponding EPR, in order to activate the subscription (transition Subscribe). Participants are also allowed to resubscribe to a resource, in this case, the transition Re-Subscribe fires, which replaces the old interval and subscription lifetime with new values.

A WS-resource discovery fails when there is no published resource with the indicated tag. The transition FailDiscover has been therefore included for this purpose, but notice that it has less priority than Discover, in order to enforce the firing of Discover when there is some published resource with the indicated tag. Furthermore, a subscription can fail when the corresponding resource has expired, so we have included a FailSubscribe transition that will fire in that case.

Notification and Subscription Time-Out Modeling

Figure 5 shows the notification and subscription time-out modeling parts. Transition Notify must be fired when a subscription condition is satisfied for some pub-

lished WS-resource, which is actually enforced by assigning the maximum priority to this transition (P1). Notification conditions are here represented by intervals, i.e., the resource value must be in the interval defined by cond1, cond2 in order to send the notification. From the generic modeling viewpoint, notifications are saved into the place Notifications, and the corresponding subscriptions are removed from the place SubscriptionRegistry.

The transition ResourceExpires (Figure 3) produces one token on the place ExpiredControl, whose color is the EPR of the expired resource. This token is used to remove all the pending subscriptions to this resource in the place SubscriptionRegistry, by firing the transition RemoveSubscription until no more tokens with this EPR are available on this place. Transition Empty can then fire (it has less priority than RemoveSubscription) to remove this token on ExpiredControl. The priority assigned to Empty guarantees us that this transition must fire before a new publication is made, to avoid that this same resource be published again without this token being previously removed from this place.

Finally, the transition SubscriptionTimeout must fire when a subscription expires, removing the corresponding subscription token from the place SubscriptionRegistry and putting it on the place SubscriptionsRemoved. This firing is enforced by its high priority (P2), but notice that in the event that several of these transitions, Notify, SubscriptionTimeout and RemoveSubscription are simultaneously enabled, Notify will always win (if enabled) to any of them, so we are giving more importance to notifications than subscription time-outs or resource lifetime expiration.

PERFORMANCE ANALYSIS

In a previous paper (Valero et al., 2015) the functionality of the model without Markovian delays has been validated, by using CPN Tools. Thus, relevant properties such as deadlock freeness were checked. We now consider a quantitative analysis with CPN Tools, which allows us to make simulations and obtain statistical results from the repetition of simulations. In order to show
Figure 6: Initial Marking and Constants using for simulations.

Figure 7: Snapshot of a Simulation (I).

how the modeled P/S system behaves we have considered the initial marking shown in Figure 6, where a maximum of 15 resources, 20 participants and 10 operations are generated. The maximum simulation time is 15, and we have considered 5 different types of resource and 10 participant identifiers.

Figure 7 and 8 show the final marking obtained once the simulation time has expired, where we can see that 14 resources were published and they finally expired, 12 discover operations failed, and one subscription from the participant Id6 to resource service 3 failed. Notice that 3 subscriptions were removed upon the expiration of the corresponding resources and 3 notifications were made.

Table 2 shows the statistical results obtained from CPN Tools by repeating the simulation 1000 times with the indicated initial marking. We can see that, in average 14.922 resource instances have been published, 0.078 publications failed, 13.226 participants were able to discover their required resources, 12.106 subscriptions were made, and 0.587 re-subscriptions. Finally, 4.587 notifications have been made, the minimum being 0, and the maximum 11.

In order to study a different case study, parameters can be adapted as necessary. Specifically, the initial marking should be updated with the new information, since the model presented here is standard for the Publish/Subscribe paradigm. For instance, if we want to study a different generation rate for resources, participants and operations, then variables r1, r2 and c respectively (shown in Figure 6) can be updated.

CONCLUSIONS AND FUTURE WORK

In this paper we have presented a Prioritized-Timed Colored Petri Net modeling of the WS-resource contents-based Publish-Subscribe paradigm, in which the resources and participants are randomly generated, by using a negative exponential distribution. We have therefore established a formal framework for the modeling of WS-resource interactions, providing a set of operations that match with the standards related to WS-resource management (WSRF), publish and discovery of services (UDDI), and Web Services Notifications (WSN). The main benefit from this formalism based on PTCPNs is that we can use CPN Tools in order to simulate and analyze the systems modeled, so we can predict their behavior, and we can discover potential problems before the implementation.

This generic model can be easily modified to introduce the specific behavior of clients, who may change their behavior in accordance with the values obtained from the resources, or as response to the received notifica-
tions. However, our goal has been to provide a generic model that can now be easily applied to a great variety of systems.

As future work in this area, the model can be extended by including exponential times in other operations and extend the formalism in order to have a complete framework for the modelling of Publish/Subscribe systems with distributed resources.

REFERENCES


WEB REFERENCES


Web Services Eventing (WS-Eventing) W3C Member Submission, March, 2006 http://www.w3c.org/submission/ws-eventing

Web Services Resource Framework (WSRF), OASIS, 2006 http://docs.oasis-open.org/wsrf/

All the figures of this work can be found at the url http://dsi.uclm.es/retics/publications/2015/ESM_2015/.
Deriving the Performance indices in Product-form stochastic Petri nets: open Problems and simulation

Simonetta Balsamo, Andrea Marin, Ivan Stojic
Università Ca’ Foscari Venezia
DAIS
email: {balsamo,marin,stojic}@deis.unive.it

KEYWORDS
Stochastic Petri nets, Simulation, Product-form

ABSTRACT

Stochastic Petri nets are an important formalism used for the performance evaluation of computer and communication systems as well as other fields like bioinformatics and logistics. Despite its high flexibility and modelling power, one of the problems of quantitative analyses based on stochastic Petri nets is the state space explosion, i.e., the high cardinality reached by the state space of even a structurally small SPN. As a consequence a direct analysis of the Markovian processes underlying the models is not feasible. Product-form Petri nets are a class of stochastic Petri nets whose invariant measure can be expressed as a product of functions, each of which depends only on a marking of a single place. Nevertheless, for the effective computation of the performance indices the computation of the stationary distribution is required. In this paper we propose a classification of product-form stochastic Petri nets based on the availability of algorithms for the computation of their stationary performance indices. Moreover, in case simulation is required, we introduce two stopping criteria that exploit the product-form property of the nets.

INTRODUCTION

Stochastic Petri nets (SPNs) - Molloy (1982), Marsan et al. (1995) - are an important formalism for assessing the performances of computer and telecommunication systems. More recently, they have been used also in other domains, such as bioinformatics with the aim of modelling biochemical reactions in organic systems (see Baldan et al. (2010) for a recent survey). SPNs, as defined in Molloy (1982), are a Markovian modelling formalism, in the sense that the stochastic process underlying the marking process (the process describing evolution of the model’s state with time) is a Continuous Time Markov Chain (CTMC). Markovian formalisms are highly appreciated because they can be analysed by numerous techniques and algorithms which have been developed for studying Markov processes in the latest decades. However, SPNs share with many other high level formalisms the problem of the state space explosion. Namely, an SPN which is structurally “small” may have a huge, possibly infinite, state space. As a consequence, although in principle the standard algorithm for transient or steady-state analysis of CTMCs could be applied, in practice time and space complexity become prohibitive and problems concerning numerical stability often arise.

In order to overcome these problems, product-form models have been introduced first in queuing networks (see Jackson (1963), Baskett et al. (1975)) and then in other formalisms including SPNs as proposed in Mairesse and Nguyen (2009), Balsamo et al. (2012), Marin et al. (2012). Product-form analyses rely on the idea that the model can be decomposed into a set of interacting components. When certain conditions are satisfied, each positive recurrent state of the underlying CTMC has a stationary equilibrium probability that can be expressed as a product of equilibrium probabilities of corresponding states of model components, obtained by considering the components in isolation. Thus, product-forms allow analyses to be performed by studying isolated components, and the solution of the system of global balance equations of the CTMC underlying the model is not required anymore. Nevertheless, for SPNs there are still some problems to address before deriving the stationary performance indices. The first problem concerns identification of the reachability set of the SPN, needed to identify the set of aforementioned positive recurrent states of the underlying CTMC. In some cases (e.g., marked graphs or state machines) the problem of deciding whether a state is reachable is computationally efficient, but for general SPNs this problem is known to belong to the class of EXPSPACE problems. Moreover, structural conditions imposed on SPN models by the product-form analysis do not reduce this complexity - Haddad et al. (2013).

In this paper we review open problems concerning the effective computation of some stationary performance indices for product-form SPNs. We show that for some SPNs, at the state of the art, one has to resort to stochastic simulation in order to obtain the desired indices, and we discuss how it is possible to exploit the product-form property to define an efficient criterion to stop the simulation.
STOCHASTIC PETRI NETS

In this section we give the definition of Stochastic Petri Nets (SPNs) and introduce notation used in the paper. An SPN Molloy (1982) is a 6-tuple

$$SPN = (P, T, I(\cdot), O(\cdot), W(\cdot), m_0)$$

where $P = \{P_1, \ldots, P_{N_P}\}$ is the set of $N_P > 0$ places, $T = \{T_1, \ldots, T_{N_T}\}$ is the set of $N_T > 0$ transitions, $I : T \rightarrow N^{N_P}$ is a function associating an input vector with each transition $T_i \in T$ and $O : T \rightarrow N^{N_P}$ is a function associating an output vector with each transition, $m_0 \in N^{N_P}$ is called initial marking of the net. Function $W : T \rightarrow \mathbb{R}^+$ assigns a positive real number to each transition $T_i \in T$. A transition $T_i$ is enabled by a marking $m \in N^{N_P}$ if $m - I(T_i) \geq 0$, i.e., has only non-negative components. We define enabling degree of a transition $T_i$ in marking $m$ by $e_i(m) = \max\{k \in \mathbb{N} : m - kI(T_i) \geq 0\}$. In general, a marking $m$ enables zero, one or more transitions. Let $E(m)$ be the set of transitions enabled by marking $m$. When a transition $T_i \in E(m)$ fires, the marking changes from $m$ to $m - I(T_i) + O(T_i)$, i.e., the tokens specified by the input vector are consumed and those specified by the output vector are produced. In Markovian Petri nets (or simply SPNs), we associate an exponentially distributed random delay with each transition enabled by a marking $m$. Thus, the non-determinism on standard Petri nets is solved with the race policy among exponential distributions. We consider two firing semantics:

- **Single server semantics**: in this case a firing delay is set when the transition is first enabled and a new delay is sampled in case the same transition is enabled after a firing. In other words, the firing rate of enabled transition $T_i$ is state independent and its value is $W(T_i)$;
- **Infinite server semantics**: in this case every enabling set of tokens is processed in parallel as soon as they arrive at the input places. Each of these concurrent delays associated with transition $T_i$ are i.i.d. exponentially distributed random variables with rate $W(T_i)$. According to the race policy, this corresponds to a single server semantics in which the firing rate depends on the marking in the transition’s input places. More formally, the firing rate of transition $T_i$ in marking $m$ is $e_i(m)W(T_i)$.

Given the initial marking $m_0$, set $RS(m_0)$ is the set of all the possible markings reachable after an arbitrary number of transition firings from $m_0$. Reachability graph of an SPN has the elements of $RS(m_0)$ as nodes and the arcs connect markings which are reachable via the firing of a transition (directly reachable markings). The marking process is the stochastic process $X(t)$ associated with the evolution of the net’s marking for $t \in \mathbb{R}^+$. It can be proved that for SPNs, $X(t)$ is a Continuous Time Markov Chain (CTMC) whose transition graph structure is identical to that of the SPN’s reachability graph. The transition rates are set according to the definition of function $W(\cdot)$ and the firing semantics which is adopted. The derivation of the reachability graph of an SPN is known to belong to the class of EX-PSPACE problems. Incidence matrix $A$ of an SPN is a matrix which has a row for each place and a column for each transition. The column associated with transition $T_i$ is $I(T_i) - O(T_i)$, and represents the marking change due to firing of $T_i$. A T-invariant for an SPN is a vector $X \succ 0$ whose dimension is equal to the number of net’s transitions, such that:

$$A \cdot X = 0,$$

where $X = (x_1, \ldots, x_{N_T})^T$ and $x_i \in \mathbb{N}$. A P-invariant of the net is a vector $Y > 0$ whose dimension is equal to the number of places of the SPN, and which satisfies the following equation:

$$A^T \cdot Y = 0,$$

where $Y = (y_1, \ldots, y_{N_P})^T$ and $y_i \in \mathbb{N}$. Support of a P-invariant: $Y$ is the set of places for which corresponding components of $Y$ are nonzero, and support is minimal if there are no P-invariants with smaller (in terms of subset) support. For each minimal support there is a unique P-invariant called minimal support P-invariant; set of minimal support P-invariants forms a basis for all P-invariants of an SPN.

**Some performance indices in SPNs**

In this part we review some performance measures for SPNs in equilibrium. Henceforth we assume that the CTMCs underling the SPNs we consider are ergodic. Let $\pi$ be the function that assigns its equilibrium probability to each positive recurrent state of the CTMC underlying the SPN. Then, the expected number of tokens in place $P_i$ in steady-state is given by the following expression:

$$\bar{N}_{P_i} = \sum_m \pi(m)m_i, 1 \leq i \leq N_P. \tag{1}$$

The throughput of a single server transition $T_i$ in steady-state is given by:

$$\bar{X}_{T_i} = \sum_m \pi(m)\delta_{e_i(m)} \geq 0 W(T_i), \tag{2}$$

with $1 \leq i \leq N_T$ and $\delta$ is the indicator function. For infinite server semantics the throughput is given by:

$$\bar{X}_{T_i} = \sum_m \pi(m)e_i(m)W(T_i). \tag{3}$$
Product-form stochastic Petri nets

A subclass of SPNs are known to be in product-form, i.e., the expression of the equilibrium distribution of the net’s marking process is such that:

$$\pi(m) = \frac{1}{G} \prod_{i=1}^{N_P} g_i(m_i),$$  \hspace{1cm} (4)

where $m$ is a positive recurrent state of the CTMC, $\pi$ is the equilibrium probability function, $G$ is the normalising constant such that the probabilities sum to 1, and $g_i$ are some positive real functions.

For the sake of simplicity, we briefly review the results on product-form SPNs only for nets whose transitions have the single server semantics. According to Coleman et al. (1996) a large class of SPNs in product-form satisfies the following conditions:

1. Let $I, \mathcal{O}$ be the sets of the input and output vectors of the net transitions, respectively. Then, $I = \mathcal{O}$.

2. No two transitions have the same input vector, i.e., $i \neq k \Rightarrow I(T_i) \neq I(T_k)$. Nets which don’t satisfy this condition can be modified by considering each set of transitions that share an input vector as a compound transition, in the following manner. If more than one transition in the net has the same input vector $I(T_i)$, we replace the set $\{T_k : I(T_k) = I(T_i)\}$ of these transitions with a compound transition $T$ and set $W(T) = \sum_{k : I(T_k) = I(T_i)} W(T_k)$. Firing of the compound transition $T$ in the modified network represents firing of one of the associated original transitions in the original net. When the compound transition $T$ fires, one of the output vectors of the associated original transitions is selected probabilistically. The selection probabilities are derived based on the properties of the exponential distribution so as to preserve the underlying CTMC of the original net. Thus, we set probability $p(I(T_i), O(T_j))$ of generating the tokens specified by $O(T_j)$ when the compound transition $T$ fires to be equal to the probability of firing of the original net $T_j$ if it is one of the transitions with the input vector $I(T_i)$ fires:

$$p(I(T_i), O(T_j)) = \frac{W(T_j)}{W(T)} = \frac{W(T_j)}{\sum_{k : I(T_k) = I(T_j)} W(T_k)}.$$

The discrete-time Markov chain with state space $I$ and the above transition probabilities is called the routing process.

3. There exists an invariant measure $f : I \rightarrow \mathbb{R}^+$ of the routing process such that:

$$\chi(i)f(i) = \sum_{j \in I} \chi(j)f(j)p(j, i),$$

where $\chi(k) = \sum_{T : I(T) = k} W(T)$.

Theorem 1 Let $C(f)$ be the vector whose components correspond to the transitions, and let the component associated with $T_i$ be equal to $\log(f(I(T_i))/f(O(T_i)))$. Since all invariant measures of the routing process differ by a positive multiplicative constant we can simply write $C(f)$ as $C$. Then, if the equation

$$-A \begin{bmatrix} \log(y_1) \\ \vdots \\ \log(y_{N_P}) \end{bmatrix} = C$$

has a unique solution then, under ergodicity assumption, for each positive recurrent state $m$ it holds that

$$\pi(m) = \frac{1}{G} \prod_{i=1}^{N_P} y_i^{m_i},$$

where $G$ is the normalising constant.

In the literature, several other classes of SPNs in product-form have been proposed such as that based on Bouchier’s full-blocking Lazar and Robertazzi (1991), Bouchier (1994), signals in the style of G-networks Marin et al. (2012) and others Balbo et al. (2003), Balbasso and Marin (June, 2007;O).

DERIVING THE PERFORMANCE INDICES FOR PRODUCT-FORM SPNS

Although Theorem 1 gives the expression for the un-normalized equilibrium distribution for a class of SPNs, derivation of the stationary performance indices requires knowledge of the normalized equilibrium probability distribution. Therefore, the efficiency of the product-form approach strongly relies on the capability of computing the normalising constant $G$ efficiently. In this section we distinguish three classes of product-form SPNs based on applicable methods for computing the normalising constant (or possibly directly the performance indices).

Cartesian product-form SPNs

This class of product-form SPNs satisfies the property that the set $RS(m_0)$ of markings reachable from an initial marking $m_0$ is Cartesian product over places of reachable markings of places. More formally, for place indices $i \in \{1, \ldots, N_P\}$, let $S_i(m_0) = \{k : \exists m \in RS(m_0) \text{ s.t. } m_i = k\}$ be the reachable markings of places. Then, an SPN is in Cartesian product-form if it satisfies the conditions of Theorem 1 and if

$$RS(m_0) = S_1(m_0) \times S_2(m_0) \times \cdots \times S_{N_P}(m_0).$$

For this class of SPNs, we can define for each place $P_i$ a constant $G_i = \sum_{k \in S_i(m_0)} y_i^k$. Then, it is easy to see that by Equation (4) we have:

$$G = \prod_{i=1}^{N_P} G_i.$$
Figure 1: Example of Cartesian product-form SPN.

As a consequence, the performance indices can be readily derived. For those who are familiar with queueing network theory, this is the case for Jackson’s networks and G-networks - Jackson (1963), Gelenbe (1989). For this class of models, each queueing stations can have a state in \( \mathbb{N} \) and the joint process state space is \( \mathbb{N}^Q \), with \( Q \) being the number of the network’s stations. Figure 1 shows an example of this class of SPNs: customers arrive from the outside via \( T_1 \) and \( T_3 \) at places \( P_2 \) and \( P_3 \). The service of customers can cause a collision (\( T_2 \) or \( T_6 \)) causing the customers to be kept in an idle phase (\( P_4 \), \( P_5 \)) and then put newly in service (firing of transition \( T_2 \)). Each place has a marking which belongs to \( \mathbb{N} \) and the joint state space is \( \mathbb{N}^6 \).

**P-invariant reachable product-form SPNs**

This class of SPNs is characterised by the fact that deciding if a marking belongs to the reachability set can be performed in polynomial time. In fact, we have that a marking is reachable if and only if given a matrix \( \mathbf{M} \) of minimal support P-invariants, a necessary and sufficient condition for the reachability of any marking \( \mathbf{m} \) is:

\[
\mathbf{Mm} = \mathbf{Mm}_0.
\]  

(5)

This class of product-form SPNs has been introduced in Coleman et al. (1996) and further studied in Sereno and Balbo (1997). For these models, in Coleman et al. (1996) the authors propose a convolution algorithm while in Sereno and Balbo (1997) a mean value analysis algorithm is proposed. The main problem for the computation of the performance indices of this class of SPNs is how to decide if the property stated by Equation (5) holds. Indeed, in general one has to derive the reachability set of the net, derive the minimal support P-invariants and verify Equation (5) for the matrix \( \mathbf{M} \) of minimal support P-invariants; alternatively one can resort to some model-dependent proof technique. An example of P-invariant reachable SPN taken from Coleman et al. (1996) is shown in Figure 2.

**General product-form SPNs**

If the product-form SPN does not belong to any of the above mentioned classes, the only possibility for carrying out an exact analysis is to derive the whole reachability set and evaluate Equation (4) for each positive recurrent state. If the state space is finite then the normalisation of the probabilities can be performed. Unfortunately, this method tends to be prone to numerical instability problems and it is time/space expensive because it requires the construction of the whole state space whose complexity is \textsc{ExpSpace}.

**SIMULATION OF PRODUCT-FORM SPNs**

Although resorting to simulation for product-form SPNs seems to be a contradiction with the analytical properties of the product-form models, in some cases it is the only possible choice. Summing up, the conditions under which one should consider to use the simulation to study a product-form SPN are:

1. The SPN is neither Cartesian nor P-invariant reachable or proving that one of these properties holds is computationally expensive;

2. The state space of the SPN is so large to make the brute force approach infeasible. By brute force approach we mean the application of Equation (4) for each positive recurrent state and then normalising
the probabilities.

In this section we discuss two possibilities for defining a stopping criterion for the simulation of product-form SPNs and we test their performance with a simulator implemented in Java.

**Stopping criteria for the simulation of product-form SPNs**

In the computation of the stationary performance indices, it is important to choose the length of two periods: the first period is delimited by the epoch at which the transient period expires and hence we can consider the model in its stationary behaviour. The second is the minimum epoch at which the simulation can be stopped in order to have accurate estimates of the desired performance indices. We have addressed the first problem by proposing a perfect sampling approach for SPNs in Balsamo et al. (2015) but several other methods developed in the literature can be applied (e.g., the Welch’s procedure presented in Welch (1983)).

In this paper we focus on the definition of a stopping criterion in the simulation of product-form SPNs. Standard stopping criteria usually rely on carrying out the stationary simulation until a certain level of accuracy of the estimated performance measure is achieved. For instance, a typical approach is defining a confidence interval and a maximum tolerance on an average measure such as the expected number of tokens in a certain SPN’s place. Intuitively, longer simulations should decrease the variance of the estimation and hence reduce the width of the confidence intervals until the desired accuracy is reached. The drawback of this approach is that it may be difficult to apply for the estimation of the probability of rare or unlikely events.

**Matrix based stopping criterion**

In the simulation of product-form SPNs we exploit the fact that although we do not know the probability of an arbitrary reachable state (because of the unknown normalising constant), from Equation (4) we can derive exactly the ratio between probabilities of two arbitrary states. Assume that we have a finite set of reachable and ergodic markings

\[ \mathcal{U} = \{ \mathbf{m}_1, \mathbf{m}_2, \ldots, \mathbf{m}_U \}, \]

and let \( \bar{\pi}_i \) be the estimation of the stationary probability for marking \( \mathbf{m}_i \) in a stochastic simulation of the SPN. Then, we define the matrix \( \bar{\mathbf{U}} \) as follows:

\[
\bar{\mathbf{U}} = \begin{bmatrix}
\frac{\bar{\pi}_1}{\bar{\pi}_U} & \frac{\bar{\pi}_2}{\bar{\pi}_U} & \cdots & \frac{\bar{\pi}_U}{\bar{\pi}_U} \\
\frac{\bar{\pi}_1}{\bar{\pi}_U} & \frac{\bar{\pi}_2}{\bar{\pi}_U} & \cdots & \frac{\bar{\pi}_U}{\bar{\pi}_U} \\
\frac{\bar{\pi}_1}{\bar{\pi}_U} & \frac{\bar{\pi}_2}{\bar{\pi}_U} & \cdots & \frac{\bar{\pi}_U}{\bar{\pi}_U} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\bar{\pi}_1}{\bar{\pi}_U} & \frac{\bar{\pi}_2}{\bar{\pi}_U} & \cdots & \frac{\bar{\pi}_U}{\bar{\pi}_U}
\end{bmatrix},
\]

We can derive the analytical values for the ratios of \( \bar{\mathbf{U}} \) thanks to the product-form property of the SPN. Let matrix \( \mathbf{U} \) be defined as:

\[
\mathbf{U} = \begin{bmatrix}
\frac{\pi_1}{\pi_U} & \frac{\pi_2}{\pi_U} & \cdots & \frac{\pi_U}{\pi_U} \\
\frac{\pi_1}{\pi_U} & \frac{\pi_2}{\pi_U} & \cdots & \frac{\pi_U}{\pi_U} \\
\vdots & \vdots & \ddots & \vdots \\
\frac{\pi_1}{\pi_U} & \frac{\pi_2}{\pi_U} & \cdots & \frac{\pi_U}{\pi_U}
\end{bmatrix},
\]

where \( \pi_i \) denotes an invariant measure of marking \( \mathbf{m}_i \in \mathcal{U} \) obtained by Equation (4). Clearly \( \bar{\mathbf{U}} \) changes along the simulation and if the accuracy of the simulation increases we will have that at a certain point \( \bar{\mathbf{U}} \approx \mathbf{U} \). More formally, we stop the stationary simulation when:

\[
|\bar{\mathbf{U}} - \mathbf{U}| < \varepsilon,
\]

where \( \varepsilon \) is a small positive real number and \( | \cdot | \) is the Frobenius’ norm.

**Vector based stopping criterion**

Let \( \mathcal{U} \) be a finite set of positive recurrent markings. Then, for each \( \mathbf{m}_i \in \mathcal{U} \) we have:

\[
G_i = \frac{\pi_i}{\prod_{j=1}^{N_F} g_j(m_{i,j})},
\]

where \( \pi_i \) is the steady-state probability of \( \mathbf{m}_i \), \( g_j \) are the functions from the product-form expression (4) and \( m_{i,j} \) is the component associated with place \( P_j \) in marking \( \mathbf{m}_i \). Let \( \bar{\pi}_i \) be the estimation of the stationary probability for marking \( \mathbf{m}_i \) in a simulation run and let

\[
G_i = \frac{\pi_i}{\prod_{j=1}^{N_F} g_j(m_{i,j})}.
\]

The we can define the vector \( \bar{\mathbf{V}} \) as:

\[
\bar{\mathbf{V}} = [G_1, G_2, \ldots, G_U].
\]

Let \( \mathbf{G} \) be defined as follows:

\[
\mathbf{G} = \frac{\sum_{i=1}^{U} G_i}{U},
\]

and let \( \mathbf{V} \) be a \( U \)-dimensional vector defined as:

\[
\mathbf{V} = [\mathbf{G}, \ldots, \mathbf{G}].
\]

Then the stopping criterion is:

\[
|\bar{\mathbf{V}} - \mathbf{V}| < \varepsilon,
\]

where \( \varepsilon \) is a small positive real number and \( | \cdot | \) is the Frobenius’ norm.
Determining set $\mathcal{U}$

Set $\mathcal{U}$ can be determined in several ways. Clearly, if we are interested in evaluating the stationary probabilities of some specific markings (e.g., because they are associated with some interesting event), we include those markings in $\mathcal{U}$. However, in general we are interested in including in $\mathcal{U}$ markings with high probability mass. In order to achieve this we perform a random walk in the CTMC underlying the SPN and we keep track of the $\mathcal{U}$ reachable markings with highest probabilities. Notice that since the SPN is in product-form, once we visit a marking we can evaluate its stationary probability modulo a multiplicative positive constant since we know its unnormalised stationary probability. Hence, we can compare the stationary probabilities of the reached markings to those with highest stationary probabilities.

**EXPERIMENTS**

In this section we present the outcomes of the simulations for some product-form stochastic Petri nets. The purpose is to validate the stopping criteria introduced in the previous section. In order to achieve this goal we apply the vector and matrix based stopping criteria to different nets which belongs to the class of Cartesian or P-invariant reachable product-form SPNs or whose state space is tractable with the brute force approach. Then, we verify if the analytical values of the performance indices or state probabilities fall in a confidence interval of 95% or 99%.

Finite state space SPN with matrix stopping criterion

Let us consider the SPN depicted in Figure 3. The transition rates are set to 5, 4, 25, 2 for $T_1$, $T_2$, $T_3$ and $T_4$, respectively. The initial marking is $[6, 4]$. The net is simple and its reachability set consists of only 11 states but it will be useful to validate our approach. We set $|\mathcal{U}| = 4$ and the random walk returns the following markings:

$$\mathcal{U} = \{[1, 9], [3, 7], [0, 10], [2, 8]\}.$$  

We set $\varepsilon = 1.5 \cdot 10^{-3}$. We performed 50 independent runs where the transient phase has been removed according to the Welch’s method. In the average the simulation had to process $235 \cdot 10^3$ transition firings to reach the desired accuracy. For the markings in $\mathcal{U}$ we obtain the following values, where $\Delta_i$ represents the width of the confidence interval for the 95% confidence level:

<table>
<thead>
<tr>
<th>Marking</th>
<th>$\tilde{\pi}_i$</th>
<th>$\pi_i$</th>
<th>$\Delta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1, 9]</td>
<td>0.240037</td>
<td>0.240010</td>
<td>3.56190 $\cdot 10^{-4}$</td>
</tr>
<tr>
<td>[3, 7]</td>
<td>0.038401</td>
<td>0.038401</td>
<td>4.70516 $\cdot 10^{-4}$</td>
</tr>
<tr>
<td>[0, 10]</td>
<td>0.600029</td>
<td>0.600025</td>
<td>2.37797 $\cdot 10^{-4}$</td>
</tr>
<tr>
<td>[2, 8]</td>
<td>0.0960075</td>
<td>0.096004</td>
<td>3.95243 $\cdot 10^{-4}$</td>
</tr>
</tbody>
</table>

We observe that all the analytical values of the marking probabilities fall in the confidence interval even though it is very small.

**Infinite state space SPN with matrix stopping criterion**

We consider the Cartesian product-form SPN depicted in Figure 4 where the rates are $1.0, 5.0, 9.0, 3.0, 7.0, 4.0, 1.7, 3.8$ for the transitions $T_1, \ldots, T_8$. The initial marking is $[1, 1, 0, 1, 0]$ and $|\mathcal{U}| = 4$. We perform 50 independent simulation runs. The algorithm constructing set $\mathcal{U}$ returns the following markings:

$$\mathcal{U} = \{[0, 0, 0, 0, 0], [0, 0, 0, 0, 1], [0, 0, 0, 1, 0], [1, 0, 0, 0, 0]\}.$$  

By setting $\varepsilon = 3 \cdot 10^{-3}$ and a confidence level of 95% we obtain the following estimates for the stationary state probabilities:

<table>
<thead>
<tr>
<th>Marking</th>
<th>$\tilde{\pi}_i$</th>
<th>$\pi_i$</th>
<th>$\Delta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0, 0, 0, 0, 0]</td>
<td>0.011779</td>
<td>0.011772</td>
<td>4.3 $\cdot 10^{-5}$</td>
</tr>
<tr>
<td>[0, 0, 0, 0, 1]</td>
<td>0.006931</td>
<td>0.006924</td>
<td>2.3 $\cdot 10^{-5}$</td>
</tr>
<tr>
<td>[0, 0, 0, 1, 0]</td>
<td>0.007235</td>
<td>0.007228</td>
<td>2.7 $\cdot 10^{-5}$</td>
</tr>
<tr>
<td>[1, 0, 0, 0, 0]</td>
<td>0.007070</td>
<td>0.007063</td>
<td>2.5 $\cdot 10^{-5}$</td>
</tr>
</tbody>
</table>

Finite state space SPN with vector stopping criterion

Let us consider again the SPN depicted in Figure 3. In this case the rates are $10, 1, 25, 2$, for $T_1, \ldots, T_4$, respectively and the initial marking is $[5, 3]$. We set $|\mathcal{U}| = 4$ and the random walk returns the following markings:

$$\mathcal{U} = \{[3, 5], [1, 7], [0, 8], [2, 6]\}.$$  

We performed 50 independent runs where $\varepsilon = 1 \cdot 10^{-6}$. The expected number of processed events to reach the
desired accuracy are $485 \cdot 10^3$. We obtain the following estimates for the stationary probabilities for a confidence level of 99%:

<table>
<thead>
<tr>
<th>Marking</th>
<th>$\tilde{\pi}_i$</th>
<th>$\pi_i$</th>
<th>$\Delta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[3, 5]</td>
<td>0.006399</td>
<td>0.00640</td>
<td>$4 \cdot 10^{-6}$</td>
</tr>
<tr>
<td>[1, 7]</td>
<td>0.16005</td>
<td>0.16</td>
<td>$6.1 \cdot 10^{-5}$</td>
</tr>
<tr>
<td>[0, 8]</td>
<td>0.79997</td>
<td>0.8</td>
<td>$5.5 \cdot 10^{-5}$</td>
</tr>
<tr>
<td>[2, 6]</td>
<td>0.031996</td>
<td>0.032</td>
<td>$1.3 \cdot 10^{-5}$</td>
</tr>
</tbody>
</table>

Also in this case all the analytical values for the stationary state probabilities fall in the confidence interval.

**Infinite state space SPN with vector stopping criterion**

We consider again the SPN of Figure 4 with transition rates 1.0, 5.0, 9.0, 3.0, 7.0, 4.0, 1.7, 3.8 for $T_1$, ..., $T_8$, respectively. We set $\varepsilon = 2 \cdot 10^{-8}$ and $|\mathcal{U}| = 11$. The initial marking is $[1, 1, 0, 1, 0]$. The markings identified by the random walk algorithm are:

$$\mathcal{U} = \{[0, 0, 1, 1, 0], [0, 0, 0, 0, 0], [0, 0, 0, 1, 0], [1, 0, 0, 1, 0], [2, 0, 0, 0, 0], [0, 0, 2, 0], [0, 0, 1, 0, 0], [0, 0, 0, 1, 1], [0, 1, 0, 0, 0], [1, 0, 0, 0, 0], [0, 0, 0, 0, 1]\}$$

We obtain the following estimates for the stationary probabilities for a confidence level of 95%:

<table>
<thead>
<tr>
<th>Marking</th>
<th>$\tilde{\pi}_i$</th>
<th>$\pi_i$</th>
<th>$\Delta_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>[0, 0, 1, 1, 0]</td>
<td>0.004216</td>
<td>0.004216</td>
<td>$1.1E - 5$</td>
</tr>
<tr>
<td>[0, 0, 0, 0, 0]</td>
<td>0.011756</td>
<td>0.011772</td>
<td>$3 - 9E - 5$</td>
</tr>
<tr>
<td>[0, 0, 0, 1, 0]</td>
<td>0.007238</td>
<td>0.007228</td>
<td>$2.4E - 5$</td>
</tr>
<tr>
<td>[1, 0, 0, 1, 0]</td>
<td>0.004339</td>
<td>0.004337</td>
<td>$1.2E - 5$</td>
</tr>
<tr>
<td>[2, 0, 0, 0, 0]</td>
<td>0.004237</td>
<td>0.004238</td>
<td>$1.2E - 5$</td>
</tr>
<tr>
<td>[0, 0, 0, 2, 0]</td>
<td>0.004439</td>
<td>0.004438</td>
<td>$1.3E - 5$</td>
</tr>
<tr>
<td>[0, 0, 1, 0, 0]</td>
<td>0.0006871</td>
<td>0.0006867</td>
<td>$2.3E - 5$</td>
</tr>
<tr>
<td>[0, 0, 0, 1, 1]</td>
<td>0.004254</td>
<td>0.004252</td>
<td>$1.2E - 5$</td>
</tr>
<tr>
<td>[0, 1, 0, 0, 0]</td>
<td>0.0006534</td>
<td>0.0006540</td>
<td>$2.4E - 5$</td>
</tr>
<tr>
<td>[1, 0, 0, 0, 0]</td>
<td>0.0007067</td>
<td>0.0007063</td>
<td>$2.4E - 5$</td>
</tr>
<tr>
<td>[0, 0, 0, 0, 1]</td>
<td>0.0006914</td>
<td>0.0006924</td>
<td>$2.1E - 5$</td>
</tr>
</tbody>
</table>

Also in this case the analytical values for the stationary state probabilities fall in the confidence intervals.

**Expected number of tokens in a place for infinite state space SPN**

In this section we study again the SPN of Figure 4 but we are interested in estimating the expected number of tokens in $P_5$ rather than the stationary probability for a marking. We compare the accuracy of the estimation with that provided by Timenet Zimmermann et al. (2000). The transition rates are 1.5, 3.0, 11.0, 19.0, 9.0, 2.0, 1.6, 3.0 for the transitions $T_1$, ..., $T_8$, respectively. The initial marking is $[1, 1, 0, 1, 0]$. We choose $|\mathcal{U}| = 4$. The analytical value for the expected number of tokens in $P_5$ is 15. We execute different tests in order to have an approximate relative width of the confidence interval of 5%, 4%, 3%, 2% and 1%. We used the matrix based stopping criterion where the values for $\varepsilon$ have been set in order to obtain the same relative width of the confidence interval. As for the product-form simulation we obtain the following table:

<table>
<thead>
<tr>
<th>Rel. width</th>
<th>Estimate</th>
<th>Rel. Err.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05297</td>
<td>15.127683</td>
<td>0.85%</td>
</tr>
<tr>
<td>0.04636</td>
<td>14.893279</td>
<td>0.71%</td>
</tr>
<tr>
<td>0.03384</td>
<td>14.930198</td>
<td>0.46%</td>
</tr>
<tr>
<td>0.02559</td>
<td>14.997131</td>
<td>0.02%</td>
</tr>
<tr>
<td>0.01620</td>
<td>15.002821</td>
<td>0.018%</td>
</tr>
</tbody>
</table>

As for the simulation estimates obtain with Timenet we have the following outcomes:

<table>
<thead>
<tr>
<th>Rel. width</th>
<th>Estimate</th>
<th>Rel. Err.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.05032</td>
<td>15.188676</td>
<td>1.25%</td>
</tr>
<tr>
<td>0.04213</td>
<td>15.156485</td>
<td>1.04%</td>
</tr>
<tr>
<td>0.03383</td>
<td>15.097450</td>
<td>0.65%</td>
</tr>
<tr>
<td>0.02732</td>
<td>15.006833</td>
<td>0.04%</td>
</tr>
<tr>
<td>0.01187</td>
<td>15.002978</td>
<td>0.019%</td>
</tr>
</tbody>
</table>

We observe that the relative error obtained with the stopping criterion based on the product-form property of the SPN is always lower than the relative error obtained with stopping criteria applicable for general SPNs.

**CONCLUSIONS**

In this paper we have reviewed the problems concerning the effective computation of the performance indices or the stationary state probabilities in product-form stochastic Petri nets. Indeed, although this class of models admits a separable solution for the stationary distribution that potentially allows for an analytical tractability of the performance measures, the problem of the efficient computation of the normalising constant is still open. We identified two classes of product-form SPNs for which determining the normalising constant
and hence the stationary performance indices is computationally feasible. For the product-form SPNs that do not belong to this class, if the state space is too large for brute-force normalisation of the probabilities, one has to resort to simulation. To the best of our knowledge, at the state of the art, there does not exist any algorithm that exploits the product-form property of the SPN in its simulation. In this paper we have proposed to use this property in the definition of two criteria for stopping the simulation. We have validated the two criteria on some SPNs and compared the accuracy of the estimates with those obtained by using halting criteria for general SPN. We showed that the proposed criteria improve the accuracy of the estimates. Future works include the application of the proposed approach to estimate the performances of sampling and game-theory based algorithms Albarelli et al. (2011), Torsello et al. (2011).

REFERENCES


SIMULATION TOOLS
ASSISI PLAYGROUND: A SIMULATION TOOL FOR COLLECTIVE BIO-HYBRID SYSTEM RESEARCH

Pedro Mariano  
Rob Mills  
Luís Correia  
BioISI – Biosystems & Integrative Sciences Institute  
Faculty of Sciences  
Universidade de Lisboa  
Campinho Grande, 1749-016 Lisboa, Portugal  
email: plmariano@ciencias.ulisboa.pt

Danjan Miklić  
Stjepan Bogdan  
LARICS Laboratory  
Faculty of Electrical Engineering and Computing  
University of Zagreb  
Unska 3, 10000 Zagreb  
Croatia  
email: danjan.miklic@fer.hr

KEYWORDS  
Simulation, Collective adaptive systems, Bio-hybrid systems

ABSTRACT

This paper presents ASSISI Playground, a simulator for facilitating research in the area of collective bio-hybrid systems. Its development is motivated by the specific use-case of simulating bio-hybrid societies consisting of honeybees and static robotic units called CASUs (Combined Actuator-Sensor Units). However, due to its modular design, the simulator can easily be extended to other animal species and other types of robot. The distinguishing features of the software are the ability to simulate societies consisting of hundreds of individuals in real-time, behaviour of individuals implemented in Python scripts that can be easily modified and extended by the user, and the ability to directly transfer controllers from simulated to real robots. Furthermore, the simulator implements several modalities of physical interaction that are not typically provided by conventional simulation frameworks but highly relevant to bio-hybrid research, including vibrations, airflow and heat transfer. In the paper, we describe the simulator architecture, provide implementation details for the physical interaction modalities and present two examples which demonstrate the usability of the simulator.

INTRODUCTION

Nature abounds with examples of complex adaptive behaviours of large animal groups, which are produced by simple, repeated interactions between individuals. Examples of such behaviours include bird flocking, fish school migration, ant pheromone trail networks, termite mounds, cockroach aggregation and honeybee foraging. Even human crowds can exhibit such emergent behaviours. For decades, scientists have been working on identifying the underlying principles of self-organization (Sumpter 2006). The importance of understanding animal self-organization is twofold. Firstly, many of the animals exhibiting such behaviour have great economic importance. While swarms of locusts and ant colonies can inflict significant economic damage, our food supply depends on cattle herds and fish schools, as well as on colonies of honeybees and other pollinating insects, which are necessary for the production of 87 out of 115 leading global food crops (Klein et al. 2007). Secondly, the principles of self-organization can potentially be applicable to a wide variety of complex technical systems which consist of a large number of interacting units, such as the generation and distribution of electrical power or traffic networks. Ant colony optimization (Dorigo and Blum 2005) is one very successful example of a technical solution that has been inspired by collective animal behaviour.

Until recently, research on self-organization in collective animal systems has been largely limited to observation. The shortcoming of this methodology lies in the fact that the behaviour of individuals is dependent on the behaviour of their interaction partners. Until one is able to manipulate the behaviour of individuals, it can be difficult to obtain quantifiable results and verify scientific hypotheses. Fortunately, advances in robotics technology are enabling revolutionary new tools for the study of social behaviour in animals. Interactive robots can exhibit standardized behaviours, they enable the decoupling of morphology from behaviour, and can implement complex interaction sequences and embody behavioural models (Krause et al. 2011). Some research groups have already been successful in creating robots which have been accepted by animals as conspecifics (Halloy et al. 2007). Our project ASSISIbf (Animal and robot Societies Self-organise and Integrate by Social Interaction – bees and fish) aims to create mixed societies of honeybees and robots, as well as fish and robots, with a very high level of social integration. We have created specialized static robotic units, called Combined Actuator-Sensor Unit (CASU) (Gripagic et al. 2015), capable of producing and sensing a variety of stimuli relevant for honeybee behaviour. By using multiple stimuli as com-
munication channels, we hope to achieve effectively one single, integrated society, a social cyborg (Schnickl et al. 2013). We expect that such a bio-hybrid society can collectively perform tasks with greater efficiency and robustness than the individual biological and robotic societies. Creating and experimenting with bio-hybrid societies is inherently a challenging and time-consuming task. In particular, researchers face the following specific challenges:

- Experimenting with animals is time consuming, and can be subject to seasonal constraints (e.g., honeybees are typically active only from late April until early September)
- Debugging algorithms on robotic hardware can be impractical and time consuming
- Solving some complex problems requires lengthy optimization procedures, such as evolutionary algorithms, which are impractical or impossible to run in an experimental setup with live animals

Computer simulation presents itself as a necessary tool for overcoming the difficulties mentioned above. End users require high-speed, qualitatively correct simulations of large numbers of individual entities, subject to a rich set of physical interactions. In addition to simple mechanical interaction (collision detection), research conducted within the ASSISIbf project requires modeling of heat diffusion, vibration propagation and airflow. Furthermore, the simulator should make it simple to implement and experiment with different behaviours of simulated entities. Transferring simulated behaviours to hardware devices should be possible without any code modifications. Due to the obvious importance of simulation tools in robotics and collective systems research, there is a wide variety of simulators available, either as commercial products or as open source projects. A detailed overview of the most popular simulators is provided by Staranowicz and Mariottini (2011). Two other interesting tools are NetLogo (Wilensky 1999), a programmable modeling environment for simulating natural and social phenomena, and ARGoS (Pincirolli et al. 2012), a simulator specifically aimed at simulating multi-robot systems. However, none of the available tools were able to cover all of our requirements. In particular, we were not able to find a tool that can provide simple and efficient simulation of all stimuli relevant for honeybee interaction, such as heat and airflow, while at the same time providing the ability to run the same code in simulation and on hardware. Therefore, we have developed a novel tool, ASSISI Playground, that we describe in this paper. Our simulator is built on top of the Enki open source simulation engine (Magnenat et al. 2009).

In the following sections, we outline the simulator architecture, provide implementation details of the implemented physical modalities and present two illustrative examples of bio-hybrid society simulation.

SIMULATION FRAMEWORK

**Simulator requirements**

The primary motivation for the development of ASSISI Playground is providing a tool to support our research on bio-hybrid societies of bees and CASU devices within the ASSISIbf project. Figure 1 shows our experimental setup. However, when designing the simulator, we have set ourselves a more ambitious design goal of providing a modular and extensible tool that can be useful in the general context of simulating collective bio-hybrid systems. Combining this overall design goal with our specific needs has yielded the following set of requirements:

1. Real-time simulation of large societies comprising hundreds of individuals
2. Scriptable behaviours of individuals
3. Calibrated models of different physical interactions: collisions, heat, vibration, airflow, object proximity
4. Transition from simulation to hardware without code modifications
5. High-speed headless simulation mode for tuning evolutionary algorithms

**Simulator design**

A high-level overview of the different system components is shown in Figure 2. The key design choice responsible for system modularity is the middleware layer built on top of the ZeroMQ messaging protocol

---

1[^1]: http://zeromq.org/

---

Figure 1: Honeybees in the CASU array.
Protobuf message serialization\(^2\). Using a messaging protocol enables the decoupling of the simulator engine from “controller code” which implements the behaviour of simulated entities. Sensor data and actuator commands are published through the middleware, and made available to the user through a Python API. This approach has been popularized in the robotics community by the ROS middleware (Quigley et al. 2009), and its advantages are twofold. Firstly, the user has full control over the behaviour of the simulated entities. For instance, different models of honeybee behaviour can be quickly implemented, tested and refined. Furthermore, because the simulated CASU devices use the same communication protocol as the real devices, code implementing CASU behaviour can be directly ported to hardware. This speeds up the development and debugging cycle significantly.

![Software architecture overview](image)

The simulation engine extends the Enki simulator with additional physical modalities relevant for honeybee behaviour, as described in the following section.

**PHYSICAL MODALITIES**

In this section we describe the physical modalities that are currently implemented in the ASSISI Playground. Specifically, we have implemented heat, vibration and airflow physical modalities because of their responses in honeybees. The simulated CASU has the following modalities: a heat actuator that models the CASU Peltier which is made of copper and has a ring shape; six heat sensors equally spaced around the CASU; six proximity sensors for detecting other objects; a single vibration actuator located in the center; six vibration sensors equally spaced around the CASU; a single air pump with six exits. See also Griparic et al. (2015) which describes the physical design of CASUs. The simulated honeybee has the following modalities: a single heat sensor located in the center; a single air flow sensor located in the center; five proximity sensors for detecting other objects.

**Heat**

The heat model is based on 2D matrix. Each cell represents a square area in the world, and so has a variable temperature. Heat flows from warmer cells to cooler cells, and can also directly flow to the outside world. Thermal flow only occurs between a cell and its Von Neumann neighbours. The update rule is:

\[
h_{i}^{t+1} = h_{i}^{t} + \alpha \left( (H_{W} - h_{i}^{t})D_{W} + \sum_{j \in N_{i}} (h_{j}^{t} - h_{i}^{t})d_{j} \right), \tag{1}
\]

where \(h_{i}^{t}\) is the heat in cell \(i\) at time step \(t\), \(N_{i}\) is the set containing the neighbouring cells of \(i\), \(d_{j}\) is the thermal diffusivity of cell \(j\), \(H_{W}\) is the outside temperature, \(D_{W}\) is the heat dissipation. The factor \((H_{W} - h_{i}^{t})D_{W}\) represents heat flow to the outside world. Border cells are held at a temperature of \(H_{W}\) to model an infinite reservoir. This means that if there are no heat sources in the world, all cells will tend to \(H_{W}\).

Parameter \(\alpha\) depends on the cell size, and on time. When using bigger cells, the changes in temperature are slower. Temperature is updated at constant intervals. More specifically, Enki has a function responsible for updating the state of the world, which has a time step parameter. When this parameter is higher, more heat is transferred in one simulation time step. In order to avoid oscillations in equation (1), \(\alpha\) cannot be greater than 1/4: larger values of \(\alpha\) would result in \(\Delta h_{i}\) being bigger than the temperature difference between two adjacent cells. The value of \(\alpha\) is:

\[
\alpha = \frac{\Delta t}{s^2}, \tag{2}
\]

where \(\Delta t\) is the simulation time step, and \(s\) is cell size.

**Heat actuator**

The heat actuators are characterised by a mesh of point sources with a thermal response time \(\zeta\). When a heat source is turned on, the cells where it is located start heating towards the target temperature. The equation used to compute the cell temperature is:

\[
fH_{T} - (1 - f)h_{i}, \tag{3}
\]
where $f = \min(1.0, \zeta \Delta t)$ is a factor that depends on the thermal response time and the simulation time step $\Delta t$, and parameter $H_T$ is the set temperature of the actuator.

A CASU peltier is represented by a circumference of point sources. The points are equally spaced along the circumference, and, depending on the grid scale, each point may or may not correspond to a single cell. A fine grid scale increases realism but trades off with simulation cost (the computational effort decreases quadratically with increasing cell size).

**Temperature sensor**

A heat sensor is characterised by its location. It senses the temperature in the corresponding cell within the heat mesh.

Figure 3 shows a screenshot from the simulator with 25 CASUs, which are heating to target temperatures from the set \{34, 40\}°C. The simulator visualisation is showing the heat layer. The cells where the CASU peltier is located have the thermal diffusivity of copper. Other cells have the thermal diffusivity of air. These different thermal diffusivity parameters, and the direct heat loss in cells, together mean that heat does not propagate so far from CASUs.

**Heat model calibration**

The model presented in this section has parameters that must be fine-tuned in order to correctly model a specific electronic device. We have run an experiment both in simulation and on the physical hardware, in which a single CASU periodically changes its set temperature in the sequence \{42, 26, 38, 30, 34\}°C. Figure 4 shows the temperature measured in both setups. Table 1 provides the parameters that best match the data from a real CASU.

<table>
<thead>
<tr>
<th>parameter</th>
<th>description</th>
<th>value</th>
</tr>
</thead>
<tbody>
<tr>
<td>$s$</td>
<td>cell size</td>
<td>0.5cm</td>
</tr>
<tr>
<td>$H_W$</td>
<td>outside temperature</td>
<td>23°C</td>
</tr>
<tr>
<td>$D_W$</td>
<td>heat dissipation</td>
<td>$10^{-6}$</td>
</tr>
<tr>
<td>$\Delta t$</td>
<td>simulation time step</td>
<td>0.1</td>
</tr>
<tr>
<td>$\zeta$</td>
<td>thermal response time</td>
<td>0.3</td>
</tr>
<tr>
<td>radius</td>
<td>radius of peltier plate</td>
<td>1.6cm</td>
</tr>
<tr>
<td>numberPoints</td>
<td>number of points in circumference</td>
<td>16</td>
</tr>
</tbody>
</table>

Table 1: Heat model parameters

**Proximity sensing**

Phenomenologically, the honeybees are able to sense conspecifics and other objects in their environment at close proximity. We model this ability through infrared proximity sensors. We use the implementation of infra-red sensors in Enki as the underlying model of a proximity sensor. Table 2 shows the sensor model parameters. Note that each sensor has a height meaning that objects lower that this value are not perceived by the sensor. The sensor reading is based on an inverse square response function and three casted rays. Each ray is separated by an angle of 15 degrees. For each ray we compute a value given by:

$$F(x) = \begin{cases} 
0 & \text{if } x > r_g \\
\frac{m}{x^2 - 2x_0 x + c} & \text{if } x < x_0 \\
\frac{m(c - x_0^2)}{x^2 - 2x_0 x + c} & \text{otherwise}
\end{cases}$$

where $x$ is the distance from the sensor position to the nearest obstacle, and other parameters are described in table 2. The values of each ray are combined in order
Table 2: Proximity sensor model parameters

<table>
<thead>
<tr>
<th>parameter</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>pos</td>
<td>Relative position on the robot</td>
</tr>
<tr>
<td>height</td>
<td>Height above ground, the sensor will not see any object of smaller height</td>
</tr>
<tr>
<td>orientation</td>
<td>Relative orientation on the robot</td>
</tr>
<tr>
<td>rg</td>
<td>Actual detection range</td>
</tr>
<tr>
<td>aperture</td>
<td>Sensor aperture angle</td>
</tr>
<tr>
<td>$\alpha$</td>
<td>Inverse of aperture</td>
</tr>
<tr>
<td>$m$</td>
<td>Maximum possible response value, might be inside the robot if $x_0 &lt; 0$</td>
</tr>
<tr>
<td>$x_0$</td>
<td>Position of the maximum of response (might be negative, inside the robot)</td>
</tr>
<tr>
<td>$c$</td>
<td>Third parameter of response function</td>
</tr>
<tr>
<td>noiseSd</td>
<td>Standard deviation of Gaussian noise in the response space</td>
</tr>
</tbody>
</table>

Table 3: Vibration actuator simulation model parameters

<table>
<thead>
<tr>
<th>parameter</th>
<th>description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c$</td>
<td>Quadratic decay</td>
</tr>
<tr>
<td>$A$</td>
<td>Maximum produced amplitude</td>
</tr>
<tr>
<td>$f$</td>
<td>Frequency</td>
</tr>
<tr>
<td>$v$</td>
<td>Vibration speed, equal to speed of sound in copper</td>
</tr>
<tr>
<td>$\theta$</td>
<td>Phase</td>
</tr>
<tr>
<td>noise</td>
<td>Frequency error</td>
</tr>
</tbody>
</table>

travels in a 2 dimensional plane. $x$ is the radial distance from the CASU vibration actuator, which is considered a point source.

In addition to equation (7), the vibration actuator is characterised by its phase $\theta$ and a frequency error. The phase allows multiple CASUs to vibrate at different phases, thus providing the possibility of vibrating out of sync. The frequency error is used when setting the frequency of a CASU. This models inaccuracy in real motors. When the user sets the frequency, the real frequency is set to the requested value plus a uniform noise with maximum value given by the frequency error. Given the variety of materials present in a CASU array, and their associated sound speeds, we assume that vibration travels at the maximum of these speeds (namely, the sound velocity in copper).

To account for decay, we assume that amplitude decays quadratically with distance. This is controlled by the rate parameter $c$. The equation that models the vibration produced by a CASU is:

$$ s(x, t) = \frac{1}{1 + cx^2} A \sin \left( 2\pi f \left( t + \frac{x}{v} \right) \right). $$

(8)

Table 3 shows all parameters that control the vibration actuator’s behaviour.

Vibration Sensors or Accelerometers
Vibration sensors measure the frequency and amplitude of all vibration actuators in the world. Both measures are perturbed by a Gaussian noise with average zero and a positive standard deviation. Vibration sensors return two vectors that correspond to the spectral response.

Airflow
Airflow is used to disrupt the honeybees and can be considered as a repulsive stimulus. Using this, CASUs may be able to steer honeybees away from specific areas.

Air Pump
An air pump is characterised by an orientation, an aperture angle, an intensity, and a range. This last parameter means that an object outside the air pump range
does not sense any air flow. An air pump produces an air flow in a cone.

**Air Sensor**
An air sensor measures the air flow from all air pumps within range and within the aperture cone. Each air pump $a$ contributes with an air flow intensity vector, $i_a$. The sensed air flow is the sum of all contributing air pumps, $\sum_a i_a$.

**CASE STUDIES**

**Honeybee aggregation around heat source**

Figure 5: Screenshot showing aggregation towards end of simulation. Bees aggregate at the warmer heat source. Bee colours indicate behavioural states (yellow=exploring; blue=avoiding object; red=paused, brighter red for longer pauses).

Figure 6: Bee location over time (upper) and statistically significant collective decision. For these parameters, it does not take long before the bees form a stable aggregation that contains most of the bees most of the time.

Juvenile honeybees form aggregations in warmer parts of their environment. It appears to be a social phenomenon: only with a sufficient population density do any aggregations arise. The strength of aggregation is modulated by the local environmental temperature, with the bees exhibiting a preference for 36°C.

To illustrate aggregation formation over time, we present a group of simulated animals with a binary choice between two heat sources, implemented by CASUs emitting a fixed temperature (see Fig. 5). One CASU is set to 36°C, the other at 28°C, and the ambient is set at 27°C.

The bees use the individual-based model behaviour *Beeclust* (Schmickl et al. 2009), which has previously been demonstrated to capture the key features of the bee behaviour. In this model, the animals move randomly about the arena until they encounter a conspecific, at which time they stop for a period that depends on the local temperature. The closer to their preferred 36°C, the longer the pause. The specific implementation is described fully in Mills et al. (2015).

25 bees are initially placed centrally in the environment, and as they explore the arena and locate the warmer heat source, they begin to form an aggregation, which recruits the majority of the animals (Fig. 6, upper). We use a binomial test to formally quantify when the group has formed a significant decision at the collective level (see Halloy et al. 2007), which shows a stable aggregation is maintained after approximately 4 minutes (Fig. 6, lower).

Elsewhere we have shown that the model yields appropriate macro-level behaviour as a function of both heat source temperature and of bee group size (Mills et al. 2015). These responses are qualitatively similar to that observed in Szopek et al. (2013).

We are not presently able to quantitatively recreate results from the prior work due to the differences in heat sources used: here we use CASUs which do not generate the same temperature fields as prior equipment (most notably, the effect of each CASU is intentionally more localised). Calibration work for scenarios such as this is ongoing.

**Affecting honeybee motion with airflow**

In this illustration, we use the airflow actuators as a “virtual fence” (see, e.g., Butler et al. 2006), which will help to retain the bees within a central zone. There is a heat source in the centre, but this is insufficient to maintain an stable aggregation in this size arena (the expected time for a bee that leaves the aggregation to re-find it is a function of area to explore, which is significantly larger than in the previous case). Accordingly, the outer ring of CASUs attempt to help the bees remain nearby the inner CASU (see Fig. 7).

Since these simulations are progressing in parallel with the hardware implementation and lab testing cycle, we use this as an exploratory exercise. We use a minimal assumptions extension to the *Beeclust* model as used above: when a bee senses airflow, it attempts to get away from the agitation by turning randomly and then moving forwards. The bees also have a “refractory” period that prevents being caught in an endless stressed state.
We use a simple controller for the ring of CASUs: when any of the inward-facing proximity sensors are triggered, the airflow is switched on temporarily; however this is overridden if any of the outward-facing sensors are triggered at the same time. The rationale is only to ‘erect the fence’ for bees on the inside, while letting bees on the outside back in. We compare this against two controls: 1) constantly emitting an airflow; and 2) actuators off (these still serve as a physical obstruction).

Although we might expect that an airflow with radius sufficient to be sensed entirely across a gap would offer an impermeable fence, this is in fact not the case. Neither the interactive robots nor the constant airflow robots are sufficient to maintain all of the bees within the central zone. However, when compared against the passive CASUs, the time it takes before the distribution of animals reaches approximately uniform over the whole arena (dashed horizontal line, Fig. 8) is increased. This experiment illustrates how we can investigate experimental design and test predictions – and, in some instances, falsify them – before spending valuable laboratory time; instead, guiding us towards experiments that will offer higher chances of success.

CONCLUSIONS AND FUTURE WORK

In this paper, we have described a simulation tool aimed at supporting research of collective bio-hybrid systems. Although developed for the specific use case of honeybees and CASU devices, we believe ASSISI Playground can be a useful tool for research on other types of bio-hybrid systems. Its distinguishing features are the ability to simulate societies consisting of hundreds of individuals in real time, behaviour of individuals implemented in Python scripts that can be easily modified and extended by the user, and the ability to directly transfer controllers from simulated to real robots. The simulator also provides a rich set of physical interaction modalities: collision detection, heat transfer, infra-red proximity detection, vibrations and airflow, which have been described in detail. Furthermore, we have presented the calibration of the heat transfer model against experimentally obtained data. Two simple usage examples have been described, which involve CASU devices interacting with honeybees by modulating their individual and collective behaviour. The software is developed as an open source project, consisting of two separate modules: the simulator and the Python API. Both modules are publicly available on GitHub.⁴

The simulator is being actively developed and several improvements are the subject of our ongoing work. As soon as experimental data becomes available, we will perform calibration of vibration and airflow modalities, using a methodology similar to the one we use for calibrating the heat model. We also plan to explore the limits of simulator scalability, i.e., find the maximum number of units that can be simulated in real time. A very important goal is to exploit headless simulation mode for running evolutionary algorithms. System parameters obtained in simulation by evolution will be experimentally evaluated on the bio-hybrid society of honeybees and CASU devices. Finally, we would like to see ASSISI Playground used for supporting research on other types of bio-hybrid societies.

⁴https://github.com/laries/assisi-playground
⁵https://github.com/laries/assisi-python
Acknowledgements

This work is supported by: EU-ICT project “ASSISI” no. 601074, and by centre grant (to BioISI, ref: UID/MULTI/04046/2013), from FCT/MCTES/PIDDAC, Portugal.

REFERENCES


Halloy J.; Sempo G.; Caprari G.; Rivault C.; Asadpour M.; Tâche F.; Said I.; Durier V.; Canonge S.; Amé J.M.; et al., 2007. Social integration of robots into groups of cockroaches to control self-organized choices. Science, 318, no. 5853, 1155–1158.


SYNTHESIS OF A THREE STAGES SERVOVALVE BY AMESIM

Ina COSTIN, Nicolae VASILIU, Constantin CALINOIU, Daniela VASILIU, Marius Daniel BONTOS
University POLITEHNICA of Bucharest, Fluid Power Laboratory
313 Splaiul Independentei, Sector 6
RO 060042 Bucharest, Romania
E-mail: vasiliu@fluid-power.pub.ro

KEYWORDS
Numerical simulation, fine-tuning, high power electro hydraulic servovalves, speed governors

ABSTRACT
The paper presents the researches activities aiming to create a high flow electrohydraulic servo valve needed for the power stage of the speed governors controlling high power hydraulic turbines operating under low oil pressure. This is the common case of any Kaplan turbine, but the refurbishing of the old Francis turbines needs the same valve configuration. Design problems, simulation methods and experimental researches are shortly presented. From an industrial point of view, the main idea of the new concept is the use of high-quality industrial electrohydraulic and electronic components only, in order to obtain good performances even for a low-pressure supply. This target generated a new approach of the design by eliminating the pipes between the flow control stages. A detailed model of a high power servomechanism containing a three-stage non-linear electro hydraulic proportional servo valve was designed by simulation, taking into account the real geometry of the metering spool windows. The valve dynamic behavior was simulated with SIMULINK and AMESIM, and finally the results were compared with some preliminary laboratory measurements. The simulated and the real responses for different inputs were found in good agreement.

INTRODUCTION
The variation of speed and the power during the operation of a hydropower unit is achieved by adjusting the water flow that passes through the wicket gates. High accuracy electrohydraulic servomechanisms actuate these control elements. The architecture of the power control system directly depends on hydropower unit size, specific speed, and the required dynamic performance. The last one depends on the quality requirements of the electrical power produced by the unit. The static and the dynamic forces that appear during the hydropower unit operation are rather large. The pistons of the hydraulic cylinders have rather large diameters, while the oil pressure is usually in the range of 20...100 bar. The shutdown time of a hydropower unit in case of a damage is usually lower than 6.0 s, in order to avoid dangerous over speeds. The current requirements regarding the quality of the electrical power provided by hydropower units are extremely strict, and meeting them requires speed governors to react at less than 2 mHz. This performance condition requires very precise positioning of the hydraulic cylinders rods, keeping a reasonable stability reserve for the speed governor. For medium size hydropower units, these contradictory requirements can be satisfied by the aid of a two-stage proportional flow servo valve (Figure 1), which has critical lap two-slope flow characteristics (Figure 2).

![Fig. 1. Two stage proportional flow servo valve 4WRLE 35 (BOSCH)](image1)

Each metering land of the spool of such a flow control valve has rectangular slots (Figure 3). The servo valve dynamics (figure 4) strongly depends on the supply pressure.

![Fig. 2. Flow characteristics of the two stage proportional servo valve 4WRLE 35 (BOSCH)](image2)
A small supply pressure and a high turbine nominal output lead to a three stages servo valve. For example, the two servomotors acting the wicket gates of the KAPLAN turbines of the IRON GATES I Hydropower Station from Danube River have a diameter of 600mm, and a stroke of 1200mm. The servomotor stroking the runner blades has 3120mm diameter and a stroke of 300mm. The pressure supply is about 40bar, and the emergency shutdown occurs in about 6 seconds. This performance requires a huge oil flow supplied by two control valves with three stages. The final stage is about 200 mm diameter. In such cases, the designer needs always a proper mathematical model and detailed simulations in order to optimize such a complex architecture (Vasiliu and Calinoiu 2003).

NEW HARDWARE DESIGN

A proportional servo valve type 4WRLE 35 produced by BOSCH was chosen to control a big diameter power stage. From security reasons, this stage was included in the fail-safe control loop of the governor, by the aid of a set of disc springs controlled by an external pressure supplied by the emergency shut down section of the speed governor.

The power stage has a complex structure, which offers four flow control ways with a special design of the spool lands shown on figure 3. Using a 100 mm spool diameter, the valve typical flow overcomes 7800 l/min under a very small pressure drop (5 bar on a metering edge). The flow of the hydraulic cylinders is normally controlled by the BOSCH servo valve. A sealed position transducer is measuring continuously the spool position, turning the whole assembly into a proportional device included in the position control loop of the wicket gates. This new combination eliminates the pipes between the control stage and the power stage from the classical design. Such a way, the dynamic performances for the speed control are still very good, in spite of the very low supply pressure of the governor. An emergency signal generated by the speed governor creates in the third stage a wide connection needed to shutdown the turbine in a few seconds. However, an emergency directional flow valve is usually maintained in the hydraulic diagram of the speed governor, in parallel with the three stages servo valve, connected to the hydraulic servomotors. One end of the spool is always connected to the control oil supply system, and the other end can be quickly connected to the tank by a big cartridge valve controlled by a small two ways directional flow valve. The symbol of the new valve is presented in figure 5 in a conventional manner.

NUMERICAL SIMULATION BY AMESIM

AMESIM software (Lebrun et al., 2009), produced by LMS Company, a member of SIEMENS Group, was selected as a current simulation tool (Vasiliu and Calinoiu, 2011). This complex software offers numerous advantages: rich library of hydraulic symbols and components, which allows the authors to use existing, proven models for well-known components (valves, cylinders); ability to simulate different part of the system at different levels of complexity, which allows the authors to model different parts of the system at different levels of detail, as required. AMESIM models are fully compatible with LabVIEW for real time and Hardware-in-the-Loop simulations, can be imported in LabVIEW and connected to a real-time or HIL simulation system. The power stage of the flow valve, which has the greatest influence on the dynamic behavior of the system, was modeled in deep detail, at the physical process level, using the Hydraulic Component library of AMESIM in order to obtain access to all the internal variables. The pilot stage, the cylinders, the pressure source etc. have been modeled at a more concise level, using predesigned blocks from the AMESIM Hydraulic and Mechanical libraries. Ultimately, the filter needed to improve the valve stability has been
modeled by a transfer function, because the internal variables are not particularly important for the current simulation. The simplest AMESIM model developed by the authors for the three stages servo valve can be seen in Figure 6. Many types of simulation have been performed using this model. First, the authors have simulated a slow linear input signal (Figure 7) in order to find the steady-state characteristics of the valve. The evolution of some parameters of the system for the first simulation run are presented in Figures 8 to 16. All the attempts to change the available free design parameters of the third valve stage in order to avoid a strong instability pointed out the need of a filter 2/2 on the spool position error. This is the same situation encountered in the synthesis of some fly control system of the big missile like Arianne (Mare 2007). It comes from the very small dumping factor of the third stage.

Figure 6. AMESim simulation network for the three stages servo valve

Figure 7. Input voltage applied for finding the steady-state characteristics

Figure 8. The first stage flow evolution

Figure 9. The second stage flow evolution

Figure 10. The servo valve flow evolution

Figure 11. The third stage spool displacement evolution

Figure 12. The steady-state servo valve characteristics
The most important quality of this kind of electrohydraulic converter is the different behavior in the null region, and outside this region. The small slope of the flow curve for small frequency deviations always occurring during the turbine start is very useful for a very quick start of the hydropower unit. The big slopes allow a quick shutdown ordered by the speed governor or by the overall power station control unit.

**EXPERIMENTAL VALIDATION**

The experimental validation of the new design was performed in the fluid power laboratory of the University POLITEHNICA of Bucharest (Vasiliu et al., 2012). Different types of analog and digital servo valves were tested in the frame of a servomechanism with strong elastic load (Figures 17 and 18), in order to prepare this subsystem for setting in the hydropower station. The results obtained in the previous chapter by numerical simulation have been compared with the experimental data collected by the authors while working on a similar speed governor for the hydropower units of Ramnicu-Valcea hydropower plant (Vasiliu and Calinou, 2003). Taking into account the high speed response of the BOSCH servo valve, used as a preliminary stage in the new three stages design, the dynamic behavior of the two new governor types is nearly the same. The only difference comes from the very low pressure supply used for controlling the hydraulic turbine speed with the new three stages servo valve. The speed governors dedicated to entirely new Kaplan turbines are sized for 60 bar, in order to reduce the size of the hydraulic cylinders, and to improve the control system performance. The authors studied a possible safety function of the pilot stage of the new servo valve using the new test bench from Figure 18. The strong spring load was calibrated for pushing the big diameter spool to the fail position. Analog or digital servo valves can be used for positioning the piston acting on a disc springs box. The failsafe position dynamics was mainly studied. This is a possible additional safety improvement of the whole speed governor structure.
CONCLUSIONS

All the design, test, and identification activities of the new servo valve project pointed out that AMESim provided a strong solver and numerical core for transient simulation. As modeling a complex multi-physics system is not the main objective of engineers, it is important to have tools and interfaces, which accelerate and optimize the design. From this point of view, AMESim is a complete software perfectly adapted for model creation and deployment. The wide field of application, including electric powertrain (Negoita 2011; Vasiliu C. 2011; Muraru 2000), mobile hydraulics (Popescu 2011; Vasiliu N. 2011) is continuously extended by the users.

ACKNOWLEDGEMENTS

The main design activity regarding the new high power servo valve was performed by the first author in the frame of a PhD thesis devoted to the high power servo valves dynamics. The authors are grateful for all the technical support received from the LMS COMPANY (now - a SIEMENS business) in different manners: free licenses, free technical training for the research team members, and many other facilities. The work has been also funded by the Sectoral Operational Program Human Resources Development 2007-2013 of the Ministry of European Funds through the Financial Agreement POSDRU/159/1.5/S/132397.

REFERENCES


WEB REFERENCES

http://www.boschrexroth.com/
http://www.eaton.com/
http://www.moog.com/
http://www.parker.com
http://www.siemens.com/
http://www.dspace.com/
http://www.adwin.de/

BIOGRAPHIES

INA COSTIN graduated in mechanical engineering in 1984, and is preparing a Ph.D. thesis in the field of the dynamics of the fluid power systems in the frame of the Fluid Power Laboratory of the University POLITEHNICA of Bucharest. She is currently R&D Manager in DIPLOMATIC OLEODINAMICA SpA from Italy. She works in the field of design, modeling, simulation, and experimental identification of the electro hydraulic control systems. She is member of FPNI, ASME etc.

MARIUS D. BONTOS graduated in Environmental Engineering from University POLITEHNICA of Bucharest in 2005, and a MD in Environmental Engineering in 2007. After a stage of IT manager at Romanian Innovation Agency, he prepared a PhD research work on using SCADA systems in environmental engineering at the Fluid Power Laboratory from the Power Engineering Department of the University POLITEHNICA of Bucharest. He remained lecturer in the same faculty.
SOFTWARE ENGINEERING SIMULATION
NEW CODESIGN SOLUTIONS FOR MODELING AND PARTITIONING
PROBABILISTIC RECONFIGURABLE EMBEDDED SOFTWARE

Ines Ghribi
LISI Lab, INSAT, University of Carthage, Tunisia
E-mail: ines.ghrr@gmail.com

Riadh Ben Abdallah
Prince Sattam Bin Abdulaziz University
LISI Lab, University of Carthage, Tunisia
E-mail: r.abdallah@psau.edu.sa

Mohamed Khalgui
LISI Lab, INSAT, University de Carthage, Tunisia
SystemsControl Lab, Xidian University, China,
E-mail: khalgui.mohamed@gmail.com

Marco Platzner
Paderborn University , Germany
E-mail: platzner@upb.de

KEYWORDS
Co-design, Embedded systems, Partitioning, Probabilistic behavior.

ABSTRACT
This paper presents new techniques for modeling and partitioning of reconfigurable embedded systems. A reconfiguration is assumed to be any addition, removal or update of software tasks. We introduce a new modeling approach where the software specification is presented by Directed Acyclic Graphs (DAG) with valued edges expressing probabilistic estimations of the task executions. The probabilistic model deals with execution uncertainties of the software tasks. A new hardware/software partitioning approach is introduced later allowing to assign the new software model to hardware efficiently while considering the probabilistic estimation, inclusion/exclusion constraints and optimizing the communication costs. The partitioning approach starts by a functional partitioning phase followed by a hierarchical partitioning in order to generate an initial partitioning respecting the assumed constraints in user requirements. A final optimization phase using the Kernighan-Lin Heuristic is applied to the previous partitioning in order to minimize inter processor communications. Finally, we show through a case study the efficiency of our proposed techniques.

INTRODUCTION
Embedded systems are a combination of computer hardware and software to perform a particular function. They are widely used in many applications such as automobiles, telecommunication systems, intelligent home devices, medical equipment, and military systems. Comparing to the hardware parts, the software parts are much easier and faster to develop and modify. Thus, software is less expensive in terms of the development cost and time. Hardware, however, provides better performance. For this reason, an embedded system designer’s goal is to minimize the weighted sum of the software delay and hardware area (Cui 2012). Nowadays, current embedded systems applications present highly dynamic and non-deterministic behavior. A wide range of such problems can be fixed by reconfigurable embedded systems able to evaluate their own execution and reconfigure themselves based on their execution status (Khalgui et al. 2011). A reconfiguration scenario is assumed, in (Tan 2006), to be any addition, removal or update of tasks to insure the required flexibility of the system according to well-defined user strategies. Depicting an unpredictable or uncertain behavior in an early design phase such as the modeling leads to an efficient co-design process especially in the partitioning phase. Hardware/software co-design is the technique of designing concurrent hardware and software components of an embedded system. The hardware/software co-design process can be divided into four stages: specification, internal representation, partitioning and synthesis (Sapienza et al. 2013). Hardware-software partitioning refers to the assignment of operations to hardware or software and it is known as one of the major problems and challenges of co-design throughout the last decade (Poornima and Kumar 2013). It is an NP-complete problem.

In this paper, we are dealing with modeling and partitioning reconfigurable embedded systems having unpredictable or uncertain behavior. The execution uncertainties are described in a probabilistic manner rather than deterministic specifications. The software model is composed of probabilistic tasks where each one executes a set of elementary functions. A Directed Acyclic Graph (DAG) models each task where the vertices are functions connected with edges. The edges are valued with both probability values estimating the execution frequency of software functions and communication costs. These functions could be related with inclusion/exclusion constraints telling whether or not functions can be executed on the same processor. The inclusion constraint could occur in case of two functions sharing the same data. The exclusion occurs when two functions have contra-
dictory functional behaviors. As to the hardware architecture, it is composed of heterogeneous processors characterized with their weight and connected with bidirectional links. The partitioning of the described software model to the hardware architecture is our second concern.

We propose a three phases partitioning approach for the proposed probabilistic software model. A functional partitioning step deals with hard constraints and aims to optimize the number of processors by evaluating inclusion/exclusion constraints. The second step generates initial partitions or clusters by evaluating the most probabilistic executions of the software model. Finally, generated partitions are optimized with iterative techniques by evaluating the combination of their communication costs and their probability values. The main goal of the partitioning is to assign the software functions to the target hardware efficiently while respecting the inclusion/exclusion constraints, optimizing the traffic on the system by assessing the probabilistic estimations without overloading the processors. We consider later a case study which illustrates the relevance of our contribution.

The current paper is organized as follows: the next section describes useful background on modeling and partitioning techniques for reconfigurable embedded systems. Section III presents the formalization of unpredictable reconfigurable system. Section IV exposes the partitioning of reconfigurable system approach along with the case study before evaluating our contribution in Section V. The conclusion and perspectives are in section VI.

BACKGROUND

We expose in this section a literature review related to our work.

State Of The Art

Software Modeling Approaches

The resolution of a problem requires the definition of a model representing all the important issues related to the specific issue. There are many approaches to model software specifications. Some works use Petri Nets as a conceptual representation of how multiple tasks are coordinated (Chen et al. 2008); others use finite state machines that represent one single activity and its reaction to internally or externally triggered events (Renaux and Pottker 2012).

Other Specification models the application with a Directed Acyclic Graph (DAG) which is a generic model of a program consisting of partially ordered processes. Indeed, the functionality of a complex system with intensive data stream operations can be abstracted as a directed acyclic graph with nodes that denote the set of tasks to be executed and a set of directed edges that refer to communications between tasks. Embedded systems can also be modeled by a data flow graph (DFG) which is a directed one with vertices and edges. Vertices correspond to actors representing inputs, outputs, storage and operations. Edges represent data dependencies between nodes (Li et al. 2007).

Probabilistic Modeling of Embedded Software

Many of the probabilistic models used in software architecture evaluation are mathematical abstractions of certain aspects of the system which are relevant to an attribute of interest. Some probabilistic properties can be quantified by solving the model analytically. Probabilistic model checking (Filieri et al. 2011) is one example that is used to verify the probabilistic properties from the system model. Some examples of analytical models used in probabilistic quality evaluations are Reliability Block Diagrams (Assayad et al. 2004), Markov Chains (Kwiatkowska et al. 2011) and Fault Trees (Ren and Bechta Dugan 1998). Certain types of probabilistic models are hard to solve analytically (e.g. Petri Nets or Queuing Networks). Hence, model simulation techniques are used to obtain the desired attributes from such models. Many simulation-based evaluation schemes can be found in the performance evaluation domain (Cortellessa and Miranda 2000).

Hardware/software Partitioning

Based on the partitioning algorithm, exact and heuristic solutions can be differentiated. In the literature, the majority of partitioning algorithms are heuristics. This is due to the fact that partitioning is a hard problem and therefore, exact algorithms tend to be quite slow for bigger inputs. More specifically, most formulations of the partitioning problem are NP-hard, and the exact algorithms for them have exponential run-time. Many researchers have applied general-purpose heuristics to hardware/software partitioning. In particular, genetic algorithms have been extensively used as well as simulated annealing (Li et al. 2010). Other less popular heuristics in this group are tabu-search and greedy algorithms (Li and Shi 2008). These methods tend to be used with data oriented applications (Mishra et al. 2014). There are also some families of well-known heuristics that are usually applied to partitioning problems. Hierarchical clustering is a constructive heuristic that builds a partitioning in bottom-up fashion by grouping nodes, using closeness-functions to estimate the costs (Hou and Wolf 1996). Another range of partitioning heuristics are iterative algorithms. They start with a given partition; often a randomly generated one; and improve it iteratively by minimizing a given cost function. Kernighan-Lin heuristic is an iterative heuristic which was substantially improved by Fiduccia and Mattheyses and later by many others (Fiduccia and Mattheyses 1982). The advantage of these heuristics is the rapidity and the capability of processing large amount of data. Kernighan-Lin and hierarchical clustering produce a good trade-off
between quality of results and computation time.

**Kernighan-Lin heuristic**

One of the most promising directions in hardware/software partitioning is the application of the Kernighan-Lin (KL) heuristic. Originally, this algorithm was developed for a formulation of the circuit partitioning problem (Kernighan and Lin 1970). Its aim is to partition a graph into two parts of equal size with a minimal number of cutting edges. It is a so-called iterative improvement algorithm, meaning that it starts from an arbitrary partition, and swaps pairs of nodes in order to improve the cost of the partition. The reason for the success of the KL heuristic is that it is fast as a greedy algorithm, but it can escape from some local optima. The application of the KL heuristic in the context of hardware/software partitioning was suggested by Vahid. He extended the KL algorithm so that it optimizes an execution-time metric instead of the original cut metric. For gain calculation of moving a node x from a cluster to another using a metric, the following formula is used. \( G_x = E_x - I_x \) where: \( E_x \) is the cost of edges connecting a node x with other cluster; \( I_x \) is the cost of edges connecting a node x within its own cluster. The advantage of this heuristic is its simple yet powerful control strategy, which overcomes many local minima without using excessive moves.

**Discussion And Originality**

The originality of our work resides in two aspects: on one hand, it resides in the new modeling technique of software architecture when taking into account execution uncertainties under many design constraints. Modeling such complex systems, dealing with reconfigurable uncertain behavior has not been treated in previous related works at the best of our knowledge. On the second hand, we introduce a new partitioning approach using our proposed software modeling technique in order to affect software tasks to the hardware effectively. The main aim of our partitioning is to reduce the traffic on the system by placing the most probable traffic first.

**FORMALIZATION OF UNPREDICTABLE RECONFIGURABLE SYSTEM**

We present in this section a new modeling approach for reconfigurable embedded software.

**Reconfigurable System Architecture**

**Function**

A function is the basic entity of the software model and it executes elementary operations. This function is characterized by a set of execution constraints and characteristics. For a precise explanation of this entity a formal model is introduced in the following.

**Definition 1 (Function).**

A function \( F \) (FID, I, O, DFG, Weight) is defined as follows. (i) FID stands for Function ID and contains the function’s identifier. (ii) I contains the set of input data that will be used by \( F \). (iii) O contains the output data generated by \( F \). (iv) DFG is a data flow graph that models the internal behavior of \( F \). It is composed of elementary operations and it takes as input I and generated as output O. (v) Weight characterizes the function in term of its execution time.

Figure 1 models a function \( F \) and its internal behavior represented by a DFG. \( F \) takes \( a \) and \( b \) as input data, calculates \( a \times b - \sqrt{a \times b} + 0.5 \) and stores the result in the output O.

![Figure 1: A Function F and its internal behavior](image)

**Task**

A software task is modeled as a Directed Acyclic Graph (DAG). It is composed of a number of functions connected by valued edges. If the system is complex, a task may contain other tasks along with functions. The DAG and the task are defined as follows.

**Definition 2 (DAG).** A DAG = \{ Node, Edge \} where: (i) A node \( Nd \) represents a software function \( F \) or a software task \( T \). (ii) An edge \( Ed = (Pr, Com) \) connects two functions. i.e \( F_i \) to \( F_j \). Ed can connect a function to a task \( T \), i.e \( F_i \) to \( T \).

(iii) \( Pr \) is the probability of executing the successor of \( F_i \) \( F_j \) in the case of two functions, \( T \) in the other case.

Com is the communication cost of the link between \( F_i \) and \( F_j \) or \( F_i \) and \( T \).

**Definition 3 (Task).** A task \( T = \{ TID, DAG, Weight, Pr \} \) is defined as follows. (i) TID stands for task ID and contains the task’s identifier. (ii) DAG is a Directed Acyclic Graph that models the task. (iii) Weight is the cost of the longest execution path in the DAG of \( T \). (iv) \( Pr \) is the probability of \( T \) execution.

We assume that all tasks are reconfigurable since they can be added, removed according to the probability \( Pr \).
They can also be changed according to their edge’s probabilities. Figure 2 represents an example of a task T1 composed of five functions F11...F51 and a task T2. The edges of T1 are valued by probability/communication cost values. For example, the function F11 calls the function F21 with 0.2 probability and their communication cost is 8.

**Embedded System**

An embedded system is a set of independent tasks plus a hardware specification. For a precise explanation of the system, a formal model is introduced in the following. In Figure 3 we present an example of a hardware model.

**Definition 4** (Embedded System).

An embedded system is modeled as $S=(S_{Sw}, S_{H})$. (i) $S_{Sw}$ represents the software part of the system. It consists of a set of $M$ independent tasks. Two tasks $T_i$ and $T_j$ are independent means neither $T_i$ is a sub-task of $T_j$ nor $T_j$ is a sub-task of $T_i$. (ii) $S_{H}$ is the hardware part of the system. It is represented by a graph where the nodes are Processors $P_i$, $i \in [1..N]$ and the arcs are bidirectional links $L_{ij}$ linking $P_i$ to $P_j$. A processor $P$ is characterized by two parameters $P=(ID, Weight_{max})$ where $ID$ is the processor’s identifier and $Weight_{max}$ is the maximum load that $P$ could support. A link $L$ is characterized by two parameters $L=(ID, Th_{max})$ where $ID$ is the link identifier and $Th_{max}$ is the maximum throughput of $L$.

**Constraints**

In this section we define the constraints related to software functions. We define two main constraints related to the software functions, the inclusion and exclusion constraints. For a precise explanation of these constraints we define the following assumptions.

- **Assign-P($P_i$)** groups the functions assigned to the processor $P_i$.
- **Assign-L($L_{ij}$)** groups the communications edges between functions affected to the link $L_{ij}$.

We formalize the exclusion/inclusion constraints then as follows.

- **Exclu(F)** groups the functions that have not to be executed on the same processor with the function F. This constraint is modeled in the task representation by marking the mathematical symbol $\subset$ on the function F (Figure 5).
- **Inclu(F)** groups the functions that have to be executed on the same processor with F. This constraint is modeled by marking the mathematical symbol $\subset$ on F (Figure 6).

- $\forall F_k, F_h \in$ Assign-P($P_i$) $\Rightarrow$ $F_k \notin$ Exclu($F_h$).
- $\forall F_k, F_h / F_k \in$ Inclu($F_h$) $\Rightarrow$ $F_k$ and $F_h$ $\in$ Assign-P($P_i$).

**Case Study**

**Example**

For a precise exposure of our modeling and problems, we propose in this section an example of our new software model, we introduce later the issues that we intend...
to solve in the next section. We assume in this example that a system is composed of two tasks and a hardware target architecture. This software model is represented in Figure 7. The hardware model is presented in Figure 3. It is composed of three heterogeneous processors $P_1$, $P_2$ and $P_3$ linked receptively with $L_{12}$, $L_{23}$ and $L_{13}$. $P_1$, $Weight_{max} = 90$, $P_2$, $Weight_{max} = 100$ and $Weight_{max} = 60$. The maximal throughput of $L_{12}$. $Th_{max} = 40$, $L_{13}$. $Th_{max} = 30$ and $L_{23}$. $Th_{max} = 35$.

Figure 7: Software model of the case study

**Problems**

To apply an effective partitioning procedure on the defined system specification, the following assumptions should be considered.

- $\forall P_i \in \{ P_1, P_2, P_3 \}, P_i, Weight_{max} \geq \sum_{F_i \in Assign-P_i} F_i, Weight$
- $\forall L_{ij}, L_{ij}, Th_{max} \geq \sum_{Comin,Assign-L_{ij}} Com$ where Com is the communication assigned to $L_{ij}$.

The obtained partitioning should minimize the combination of communication cost and probability : $\forall L_{ij}$, Minimise the Communication cost $\times$ Probability / Communication $\in$ Assign-$L(L_{ij})$.

**PARTITIONING OF RECONFIGURABLE SYSTEM**

The goal of the task partitioning is to find an implementation that fulfills all the system requirements and respects the design constraints.

**Motivation**

The system we are designing is composed of a set of tasks and a hardware architecture composed of a set of processing elements (PE). The goal of task partitioning is to assign these tasks to the PE in a manner that:

(i) Minimizes the communication costs of links between PE.
(ii) Maximizes the occupation of PE. Figure 8 shows the organization chart of the proposed partitioning approach.

Figure 8: Organization chart of the partitioning approach

In our proposed methodology, we evaluate different constraints at each phase. We start with functional partitioning in order to generate a functional graph by evaluating the inclusion/exclusion constraint. This graph is composed of clusters connected to the rest of the graph. Couples that are concerned with these constraints are placed in either the same or different clusters. Since this constraint is hard, the clustered tasks are locked and can’t be moved any more. If this phase fails, it means that the hardware architecture cannot support the software application and no partitioning respecting the inclusion/exclusion constraints could be found. In the next phase, we choose to apply hierarchical clustering to cluster the remaining functions that have no inclusion/exclusion constraints. The tasks are evaluated by their probabilities and high probability values are treated first. Hierarchical clustering method is used as a major phase in partitioning because it has good characteristics for generating initial clusters. But, this algorithm is still a greedy method and may lead to local optimal solutions, since the earlier clustering decision could not predict the later partition’s structure. But the generated clusters present a good entry for iterative
algorithms as kernighan-Lin algorithm used in the next phase. If this step fails, we generate initial random clusters as an entry for the last phase. The goal of the third phase is to optimize the generated clusters from the previous phases. We choose to apply Kernighan-Lin heuristic to assess the best cluster for each task. This phase evaluates both probability and communication cost constraints by gain $G_c$ calculation. Hence, Kernighan-Lin heuristic is based on iterative improvement, but allows the partitioning process to escape from some local minimum.

To test our new partitioning methodology, we propose to partition the software specification presented in the previous section into clusters. We present the resulted clusters for each phase of the partitioning process.

**Functional Partitioning**

The first step is to evaluate the inclusion/exclusion constraint and generate initial clusters with locked functions. These clusters will hold the functions which respect the inclusion/exclusion constraints. Moreover, this step optimizes the number of generated clusters since their creation depends on the inclusion/exclusion constraints.

**Motivation**

- **Step 1:** ∀ task $T_i$ where $i \in [1..M]$; for each pair of functions $F_k$ and $F_l$ with $F_k \in \text{Inclu}(F_l)$ group $F_k$ and $F_l$ on the same cluster,

- **Step 2:** ∀ task $T_i$ where $i \in [1..N]$; for each pair of functions $F_k$ and $F_l$ with $F_k \in \text{Exclu}(F_l)$ put $F_k$ and $F_l$ on different clusters,

- **Step 3:** ∀ cluster $C_i$ and ∀ processor $P_j \in S_H$, $C_i$.Weight $\leq P_j$.Weight$_{\text{max}}$.

**Algorithm**

We expose in this section, the inclusion and exclusion algorithm.

The task $T$ is composed of $n$ functions. The algorithm evaluates each function $F_i$ to detect inclusion or inclusion constraints. We define the data structure as follows, (i) Tab : a table containing all the $F_i$ that compose $T$, (ii) CreateClusters(x): is a function that creates clusters with a size $x$ corresponding to the maximum size of the processor, (iii) $\text{Inclu}(F_i)$ is a table containing all the functions that have to be clustered with $F_i$, (iv) $\text{Exclu}(F_i)$ is a table containing all the functions that have not to be clustered with $F_i$, (v) $\text{Cluster}(C, x)$ is a function that clusters a node $x$ in the cluster $C$, (vi) $\text{GetBestCluster}(x)$ is a function that evaluates the available space of the input clusters and finds the best cluster for $x$.

**Algorithm 1 functional partitioning algorithm**

1: $\text{Tab} \leftarrow T$
2: $\text{CreateClusters}(\text{Pi}.\text{maxWeight})$
3: procedure F-PARTITION(tab)
4:     for $i \leftarrow 1, \text{Tab}.\text{size}$ do
5:         if $\text{Inclu}(\text{tab}[i]).\text{size} \neq \text{null}$ then
6:             for $j \leftarrow 1, \text{Inclu}(\text{tab}[i]).\text{size}$ do
7:                 $C = \text{GetBestCluster(}\text{tab}[i], \text{Inclu}(\text{tab}[i])[j]\text{)}$
8:                 $\text{Cluster}(C,\text{tab}[i])$
9:             $\text{Cluster}(C, \text{Inclu}(\text{tab}[i])[j])$
10:         end for
11:         lock(\text{tab}[i], \text{Inclu}(\text{tab}[i])[j])
12:     end if
13:     if $\text{Exclu}(\text{tab}[i]).\text{size} \neq \text{null}$ then
14:         for $j \leftarrow 1, \text{Exclu}(\text{tab}[i]).\text{size}$ do
15:             $C = \text{GetBestCluster(}\text{tab}[i]\text{)}$
16:             $C_c = \text{GetBestCluster(}\text{Exclu}(\text{tab}[i])[j]\text{)}$
17:             if $C \neq C_c$ then
18:                 $\text{Cluster}(C, \text{tab}[i])$
19:                 $\text{Cluster}(C_c, \text{Exclu}(\text{tab}[i])[j])$
20:             end if
21:         end for
22:         lock(\text{tab}[i], \text{Exclu}(\text{tab}[i])[j])
23:     end if
24: end procedure

**Running Example**

**Case of $T_1$:** F11 and F12 must be in the same cluster. This cluster must not contain F14.

**Case of $T_3$:** F31, F33 and F34 must be together in the same cluster away from F36 and F37. The resulted functional graph is presented in Figure 9.

![Functional Graph Of T1](image1)

![Functional Graph Of T3](image2)

**Figure 9:** Resulted Functional Graphs after applying the functional partitioning respectively to $T_1$ and $T_3$

**Hierarchical Partitioning**

This phase optimizes the communications costs on link $L_{ij}$, since it stores the most probabilistic traffic on the same processor. It also optimizes the processor occupations; it assigns tasks to the maximum load of proces-
**Motivation**

The hierarchical partitioning aims to generate initial clusters by evaluating the probability as a metric. We dispose of a functional graph generated by the functional partitioning phase. The following steps constitute the hierarchical Partitioning idea.

- **Step 1**: For each task \( T_i \) represented by its functional graph, we group in \( G \) the rest of the non-clustered functions \( F_j \) of \( T_i \).

- **Step 2**: For each Function \( F_j \in G \), we determine all its predecessors and their connecting edge’s probability.

- **Step 3**: Extract the highest edge’s probability for each \( F_j \in G \) and cluster \( (F_j) \) with its related clustered functions. This way we guarantee that the most probabilistic traffic is stored on the same processor. Hence, the link \( L_{ij} \) communicates only the less probabilistic traffic.

**Algorithm**

The non-clustered functions in the functional graph are stored in a table \( Ta \). The algorithm evaluates each function \( F_i \in Ta \) and extracts its highest connected edge probability. (i) \( Ta \) : a table containing all the \( F_i \) that have not been clustered, (ii) GetPredecessors(F) : a function that returns a table containing all the predecessor functions of F, (iii) ChooseMaxProbaEdge(F, X) : a function that returns the predecessor function of F having the highest probabilistic edge, (iv) cluster(F) : returns the cluster storing the function F. The complexity of this algorithm is \( O(n^2) \), where \( n \) is the size of \( Ta \).

**Algorithm 2 Hierarchical clustering algorithm**

1. \( C1, C2 \leftarrow \text{cluster} \)
2. \( Ta \leftarrow \text{tableOfNonClusteredFi} \)
3. **procedure** \( \text{H-PARTITION(Ta)} \)
4. for \( i \leftarrow 1, Ta.size \) do
5. \( T \leftarrow \text{GetPredecessors}(Ta[i]) \)
6. for \( j \leftarrow 1, T.size \) do
7. \( F \leftarrow \text{ChooseMaxProbaEdge}(Ta[i], T[j]) \)
8. if \( \text{cluster(F).WEIGHT} \max \) then
9. \( \text{cluster(F.cluster, Ta[i])} \)
10. end if
11. end for
12. end for
13. **end** procedure

**Running Example**

**Case of \( T_1 \):** Ta contains the non-clustered functions and tasks; \( Ta = \{F13,F15,T2\} \). For F13, \( T = \{F11\} \), we cluster then F13 with F11. For F15, \( T = \{F2,F14\} \), and the highest probabilistic edge is the one connecting F14 to F15. So we cluster then F15 with F14. For T2, \( T = \{F13,F15\} \), and the highest probabilistic edge is the one connecting F15 to T2. So we cluster then T2 with F15.

**Case of \( T_3 \):** Ta contains the non-clustered functions; \( G = \{F2,F5,F8\} \). If we choose F2 first, \( G_p = \{F1\} \), we cluster then F2 with F1. For F5, \( G_p = \{F2,F6\} \), and the highest probabilistic edge is the one connecting F5 to F6. So we cluster then F6 with F5. For F8, \( G_p = \{F4\} \), we cluster then F4 with F8. The resulted clusters are presented in Figure 10.

![Initial clusters of T1 and T3](image)

**Figure 10:** Resulted clusters after applying the hierarchical clustering algorithm respectively to functional graph of \( T_1 \) and functional graph of \( T_3 \)

**Kernighan-Lin Heuristic**

This phase aims to optimize the generated clusters from the hierarchical clustering phase by iterative improvements. It uses the communication cost along with the probability to refine the solution.

**Motivation**

Kernighan-Lin optimizes the partitions based on some metrics. In our partitioning process we use the combination of two metrics in order to optimize the traffic circulation of the system. We apply the following steps.

- **Step 1**: start with choosing an unlocked function \( F \in \text{Partition Pa} \).

- **Step 2**: calculate the gain \( G \) of moving F from a partition to another, \( G_F = (E_F \times P_e - I_F \times P_t) \) where: (i) \( E_F \) is the communication cost of edges connecting F with the other clusters, (ii) \( I_F \) is the communication cost of edges connecting F within its own cluster, (iii) \( P_e \) is the probability of edges connecting F with the other clusters, (iii) \( P_t \) is the probability of edges connecting F within its own cluster.

- **Step 3**: If \( G_F \geq 0 \) then we move F to the other cluster otherwise we leave F in its place.
Algorithm
The Kernighan-Lin algorithm described below includes two parts. The first one represents the control strategy that executes two actions: (i) selectNextMove() is a procedure that selects the next move, (ii) Terminate() is a procedure that returns the results if no improvements is possible. The second part of the algorithm is the data used for cost calculation: (i) DS is a data structure used to model nodes, (ii) UpdateData() initializes DS and updates it after any move, (iii) CostFct(): the value of this function combines the metrics considered in the algorithm (Frank and Thuy 1997).

Algorithm 3 Kernighan-Lin algorithm

\[ P \leftarrow\text{Partition}\]

\begin{algorithm}
\begin{algorithmic}
\Procedure{K\text{-}Partition}{U}
\For{i \leftarrow 1 \text{ to } n}
\State currP = bestP = P
\While{UnlockedTasksExist(currP)}
\State swap \leftarrow selectNextMove(currP)
\State currP \leftarrow MoveAndLockNodes(currP, swap)
\State bestP \leftarrow GetBetterPartition(bestP, currP)
\EndWhile
\If{not(CostFct(bestP) < CostFct(P))}
\State return P
\Else
\State P \leftarrow bestP
\State UnlockAllNodes(P)
\EndIf
\EndProcedure
\end{algorithmic}
\end{algorithm}

Running Example
Case of T1: For the first initial clusters of T1, the function and tasks that can be moved to the other cluster is F15 and T2. F15 is connected to F14 and T2 as internal communications as well as F12 and F13 as external communication. We extract from the graph the communication costs and the probabilities then we compute the gain values of moving F15 to the other cluster. Gain (F15) = \((E_{F15} \times P_{F15} - I_{F15} \times P_{F15}) = 6 - 20 = -14 \leq 0\). Then F15 is not moved to the other cluster. T2 is connected to F15 as internal communication and F13 as external communication. Gain (T2) = \((E_{T2} \times P_{T2} - I_{T2} \times P_{T2}) = 3 - 12 = -9 \leq 0\). Then T2 is not moved.

Case of T3: The functions that can be moved in the first cluster are F38 and F32. F38 has no external communication with the other cluster therefore it stays at its place. Gain (F32) = \((E_{F32} \times P_{F32} - I_{F32} \times P_{F32}) = 2.8 - 2.4 = 0.4 \geq 0\). Then F32 is moved. The functions that can be moved in the second cluster is F35.

F35 has no external communications with the other cluster since F32 is moved to the same cluster as F35. The resulted clusters after optimization are presented in Figure 11.

Figure 11: Resulted clusters after applying the Kernighan-Lin algorithm respectively to T1 and T3

EVALUATION OF PERFORMANCE

In order to evaluate the efficiency of our modeling and partitioning technique, we applied the proposed algorithm in (Poornima and Kumar 2013) on the same data set of our case study. We evaluated the communication cost (CC) of each method. In the algorithm (Poornima and Kumar 2013), execution costs are valued instead of probability \times CC in our method. The results of partitioning and communication costs applied on Task T1 and T3 with each method are compared in Table 1. Table 2 contains a comparison between the two methods when the number of functions is in increasing order.

Table 1: Comparison of results for T1 and T3

<table>
<thead>
<tr>
<th>Model</th>
<th>Proposed Algorithms</th>
<th>Referenced Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Allocation of T1</td>
<td>(F_{11}F_{12}F_{13}\rightarrow P1 ) (F_{14}F_{15}T2\rightarrow P2 )</td>
<td>(F_{11}F_{15}\rightarrow P1 ) (F_{13}T2\rightarrow P2 ) (F_{12}F_{14}\rightarrow P3 )</td>
</tr>
<tr>
<td>Allocation of T3</td>
<td>(F_{31}F_{33}F_{34}F_{35}\rightarrow P1 ) (F_{32}F_{35}F_{36}F_{37}\rightarrow P2 )</td>
<td>(F_{33}F_{37}\rightarrow P1 ) (F_{32}F_{35}F_{36}\rightarrow P2 ) (F_{34}F_{38}F_{31}\rightarrow P3 )</td>
</tr>
<tr>
<td>CC(T1)</td>
<td>39</td>
<td>57</td>
</tr>
<tr>
<td>CC(T3)</td>
<td>65</td>
<td>77</td>
</tr>
</tbody>
</table>
Table 2: Comparison of CC when the number of functions is in increasing order

<table>
<thead>
<tr>
<th>Function</th>
<th>Processors</th>
<th>Proposed Algorithms</th>
<th>Referenced Algorithm</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>3</td>
<td>39</td>
<td>57</td>
</tr>
<tr>
<td>7</td>
<td>3</td>
<td>55</td>
<td>71</td>
</tr>
<tr>
<td>8</td>
<td>3</td>
<td>65</td>
<td>77</td>
</tr>
<tr>
<td>9</td>
<td>3</td>
<td>84</td>
<td>112</td>
</tr>
<tr>
<td>10</td>
<td>3</td>
<td>106</td>
<td>123</td>
</tr>
</tbody>
</table>

Figure 12 and Figure 13 present the difference between the proposed algorithm and the referenced algorithm in terms of communication costs between PE and the traffic probability on the links when applying both partitioning approaches. It can be observed from these figures that the values of total cost and traffic probability obtained by the proposed algorithm are less as compared to those obtained in (Poornima and Kumar 2013). Thus, it is concluded that the present algorithm results into better cost and optimal traffic assignment in the case of an increasing number of tasks.

CONCLUSION AND PERSPECTIVES

In this paper, the problem of modeling and partitioning of reconfigurable embedded software under execution uncertainties is discussed. As the partitioning problem is known to be NP-hard, the proposed technique finds near optimal system cost. A combination of constructive and iterative algorithms is used to achieve this objective. The proposed algorithm tries to form clusters of functions and then allocate these clusters to the processors. The effectiveness of the proposed algorithm is compared with the algorithm proposed in (Poornima and Kumar 2013). This paper treats only the modeling and partitioning phases of embedded systems design. Real-time, memory and energy constraints will be treated in future works.

REFERENCES


MODELING ISSUES WHEN USING SIMULATION TO TEST THE PERFORMANCE OF MATHEMATICAL PROGRAMMING MODELS UNDER STOCHASTIC CONDITIONS

Anne-Laure Ladier  
Université de Lyon, INSA Lyon,  
DISP laboratory EA 4570  
Bât. L. de Vinci, 21 av. J. Capelle  
69621 Villeurbanne, France  
anne-laure.ladier@insa-lyon.fr

Allen G. Greenwood  
Poznan Univ. of Technology  
Faculty of Engineering Mgmt.  
Strzelecka 11, 60-963  
Poznan, Poland  
allen.greenwood@put.poznan.pl

Gülgün Alpan  
Univ. Grenoble Alpes, G-SCOP,  
F-38000 Grenoble, France  
CNRS, G-SCOP, F-38000  
Grenoble, France  
gulgun.alpan@g-scop.fr

KEYWORDS
Discrete-event simulation, discrete mathematical-programming optimization, validation, verification.

ABSTRACT
Discrete-event simulation (DES) models and discrete mathematical-programming optimization (DMPO) models are often used together in a variety of ways. This paper discusses the issues that modelers must address when using DES models to test the performance of DMPO models in a stochastic environment. The issues arise during validation of the simulation models – comparing the simulation results under deterministic conditions with results from deterministic optimization models. In our case, the issues are derived from validating simulation models that are used to test the performance of scheduling and resource allocation models (integer and mixed-integer programming optimization models) under various types of uncertainty. The models are from our work in crossdocking operations; however, we believe they are relevant to a wide variety of problem domains. In addition to describing the issues, we offer suggestions on how modelers might address the concerns.

INTRODUCTION
Modelers often employ both simulation and optimization models, combined or related in various ways, to address a particular problem. This paper identifies four relationships between simulation and optimization models that allow the two disparate modeling types to be combined to address a specific problem. These relationships, as illustrated in Figure 1, are defined as to whether they are recursive or not and as to which model type utilizes (or is supported by) the other.

The relationship in Panel (a) of Figure 1 is recursive with a simulation model utilizing an optimization model. An optimal decision is made within a simulation model and thus optimization supports simulation. For instance, Clausen et al. (2012) simulate the operations within a logistic network using optimization (multi-stage mixed-integer program, solved with a modified tabu search) to make decisions regarding the routing between the different terminals. In order to realize this relationship, the optimization is typically embedded within the simulation model.

Panel (b) of Figure 1 also illustrates a relationship where a simulation model utilizes an optimization model, but the relationship is non-recursive. In this case, a simulation model is used to test the results of an optimization model, e.g., a schedule. Wang and Regan (2008) propose two time-based algorithms for the inbound truck scheduling problem in a crossdock, evaluated with a detailed simulation model. Liu and Takakuwa (2010) test the inbound truck schedule and the employees’ schedule in a fresh-food crossdock operation using a simulation model. Deshpande et al. (2007) use discrete-event simulation to evaluate the performances of various heuristics for the problem of assigning trucks to the different doors of a crossdocking platform.

The relationship in Panel (c) of Figure 1 is recursive with an optimization model utilizing a simulation model. In this case, a simulation model is typically embedded within an optimization model and the simulation is used to evaluate the objective function associated with a solution obtained from the optimization model. Olafsson and Kim (2002) provide tutorials for this technique, which they refer to as “simulation optimization.” Greenwood et al. (2005) describe embedding simulation and optimization models in a decision support system to improve shipbuilding operations. In the logistics field, Ackelien and Adewumi (2006) use simulation as a black box to evaluate the objective function within a meta-heuristic for the cross dock truck-to-door assignment problem. In a different approach, Almeder and al. (2009) translate the solution of the optimization model into decision rules for the discrete-event simulation, and apply the procedure iteratively until a stable point is reached.

Figure 1: Complementary uses of simulation and optimization models
In Panel (d) of Figure 1, the relationship is non-recursive: simulation models generate data that are then used in an optimization model. For example, to address a personnel planning problem at a crossdocking center, Liu and Takakuwa (2009) use a simulation model to determine the workload needed. These data are inputs for an integer programming model which produces an optimal schedule for the operators, taking their skills into account. In another example, Hauser (2002) uses a simulation model to provide data on alternative layouts in a manufacturing plant.

The focus of this paper is on the non-recursive relationship where a simulation model utilizes an optimization model, as illustrated in Panel (b) of Figure 1. Based on our experience with developing and testing simulation and optimization models that interact in this manner, we detail and explain the modeling issues raised by such a relationship. We explain how those issues can be solved or circumvented. The goal is to provide the modeling community with useful insights on this application of simulation and optimization and to encourage and further enable the use of discrete-event simulation models as a means to assess the performance of optimization models.

Discrete mathematical programming optimization (DMPO) models can represent systems in a very realistic way, taking into account as many details as the simulation does; however, adding too many details makes the solution non-computable. Assumptions are often made in order to simplify the optimization model and focus on the most salient aspects. A discrete-event simulation (DES) model can be used to validate those assumptions and to determine their validity range. On the other hand, some simplifications can be made in the DES model in order to closely follow the assumptions made in the DMPO model. This is important in order to validate those assumptions.

To validate a model is to determine whether or not it is a meaningful and “accurate” representation of the real system, and contains sufficient accuracy to meet its intended use. It is about “building the right model.” Verification is the process of determining whether a model is working as intended. It is about “building the model right.”

In order to validate and verify the DES model, one expects it to behave similar to the DMPO model under deterministic conditions. In a second step, the DES model will be used under realistic, stochastic conditions in order to assess the performance and robustness of the DMPO schedules.

Figure 2: Case 1 – DES flow diagram, links with the DMPO model

Figure 3: Case 2 – Links between DMPO and DES models

This paper describes how, due to differences in the modeling approaches, disparities can occur even in the first step (validation and verification), when the models are developed to represent the same system in the same operating environment. The examples on which our observations are based come from the logistics domain (crossdocking operations), but we believe they can be relevant to a wide variety of problem domains. In addition to describing the issues, we offer suggestions on how modelers might address and solve them. Therefore, this article seeks to help modelers in the use of discrete-event simulation to assess the performance of mathematical optimization models.

BASES FOR IDENTIFYING MODELING ISSUES

The modeling issues defined in this paper are the result of testing, using discrete-event simulation, two optimization models for robustness under operational conditions that differ from those explicitly considered in the mathematical formulation, e.g., operating in a stochastic environment.

In the first case (referred later as “Case 1”), test schedules are obtained using the DMPO program described in Ladier and Alpan (2014). A schedule is generated for inbound and outbound trucks to a crossdocking facility that maximizes transportation providers’ satisfaction (in terms of the closeness to their desired arrival and departure times) and minimizes total quantity of items placed in temporary storage (rather than being directly loaded onto an outbound truck). The obtained schedule gives the exact arrival and departure time of the inbound and outbound trucks, as well as the detailed pallet moves inside the platform. A key assumption in the optimization model is that unloading, scanning, transfer, loading and departure operations can all be done within the same time period (e.g., 60 minutes) if the inbound truck and the outbound trucks are both present. That is, the time period is long enough to ensure pallets can be transferred to storage, or to their outbound truck, in masked time. The transfer capacity inside the platform (i.e., the quantity of pallets that can be moved at each time period) is limited. Also, the distance of the transfer (thus the location of the doors) is not taken into account. We refer the interested reader to Ladier and Alpan (2014) for more details about the DMPO model and assumptions.

A DES model is used to test the schedules’ robustness when subjected to various levels of randomness, e.g., early or late truck arrivals (modeled with exponential distributions), variations in process times (unloading and transfer, modeled using triangular distributions). Figure 2 shows a simplified flow diagram of the DES model. The diagram identifies the
sources of information from the DMPO model that are used by the DES and the sources of variability which provide the stochastic environment for the test.

Case 2 takes place in the same platform, but the focus is on the employees rather than the trucks. The truck schedule and truck door assignments are inputs of the problem. Test schedules are generated using the DMPO model described in Ladier et al. (2013), which uses three mixed-integer linear programs solved in sequence. The sequential solution processes results in detailed timetables (with 15-minute precision) for the employees of the logistics facility. The task assignments have to cover all of the workload for one day, while taking into account the employees’ competencies by assigning each of them to tasks for which they are most proficient. More details on the assumptions and the solution methods can be found in Ladier et al. (2013).

A DES model is used to test the robustness of the timetables generated by the DMPO, when subjected to randomness in the amount of workload. In this DES model, the workers are therefore explicitly represented, with their own capacities and their respective competencies. Figure 3 shows the links between the DMPO and the DES model in Case 2.

Both simulation models were developed using the simulation software FlexSim© (www.flexsim.com). Although both models address crossdocking scheduling problems, we believe that the issues raised in the next sections are not unique to logistics or crossdocking, and can be encountered in other modelers application domains.

FOUNDATIONAL DIFFERENCES

The first set of dissimilarities between DMPO and DES models include foundational differences in the ways the two modeling approaches represent the underlying system. These differences are described in terms of time representation, spatial representation, model structure, and model size.

Time representation

How the passage of time is represented in models constitutes a major difference between DMPO and DES. Temporal DMPO uses discrete time intervals where events and resulting activities occur within a time period. For example, within a one-hour time interval a truck arrives for unloading or a task is assigned to, and completed by, an employee. All that is considered is that these events/activities occur somewhere within the interval; the exact time is not important to the model. However, DES has a much finer granularity, events occur at precise instances of time; e.g., a truck arrives 27,1752 minutes after the arrival of the previous truck. Also, in simulation, events trigger, and are triggered by, other events; therefore, timing is an important element.

Because of these key differences, the behavior of a DMPO model using discrete time intervals and the behavior of a DES model will rarely be matched exactly. In Case 1, the DMPO model only allows a truck to leave at a multiple of the time interval considered, e.g. 60 minutes, while the trucks in the DES model leave when a specified condition is met, e.g., when a truck is empty (inbound) or full (outbound). Therefore, if we compare the truck departure times as calculated by the DMPO model and as observed in the DES model, we incur time differences as large as 59 minutes even though both models behave as expected. Those differences can be reduced by shortening the time intervals used in the optimization model; however, that makes the optimization model more complex (and possibly incomputable) and some differences will always be observed. One way to circumvent this issue is to measure performance in terms of intervals. For example, assuming the masked time is 60 minutes in the optimization model, then if a departure is planned at 17:00 in the optimization model and if the truck departs at 17:11 in the simulation model, then the truck departure is considered “on time” and there is no difference in the model results.

Modelers should therefore be aware of the differences in granularity of the modeling approaches and they can circumvent this issue by using time intervals rather than absolute time for their simulation measures.

Spatial representation

DES models not only consider events in time, they often consider spatial relationships among modeling elements and the effects these relationships have on system behavior and performance. Most simulation software integrate and enable the use of locational data to determine activity times; e.g., each travel time in a DES model may be based on the current location of a transporting resource, its destination(s), speeds and possibly acceleration, etc. This granularity is not always considered in optimization formulations.

DMPO models can take into account speed and acceleration – but this adds considerably to model complexity. Therefore, there is a tradeoff between fidelity in the optimization model (zero travel times) and closeness to realistic operations. As a result, spatial effects are taken into account in DMPO models only if they significantly impact the key performance measures that are used in decision making. For example, the selection of an alternative may be heavily influenced by the distance walked by employees in a crossdock facility. If spatial considerations are not at the core of the problem, then mathematical programming modelers tend to ignore travel time or use masked time in order to simplify the optimization models. An action time that is “short enough” can be considered as instantaneous, i.e., performed within the formulated time interval (Case 1).

The difference that DES models typically consider the spatial aspects of systems being modeled, and DMPO formulations do not, causes cross-model validation challenges. To mitigate this issue we propose a compromise approach: control the transfer time by making it a process step in the simulation that does not consider distances and speeds. This easily permits setting the transfer time to zero so that it can be compared to the mathematical programming model, yet enables an easy extension to the simulation model in order to incorporate more realistic aspects, such as probability distributions and location/speed considerations.
Model structure and size

Model complexity is often defined by the model structure and its size. DES models and DMPO models define model structure and size quite differently.

In DMPO, size is not of special concern in formulating or describing a model since the constraints are specified in a tight mathematical notation and input parameters are provided in a structured manner. However, the size or theoretical complexity of the problem drives the choice of solution method, and therefore the solution accuracy and speed. Optimal solutions may be found, but if the problem is NP-hard, execution time increases exponentially with the problem size. On the other hand, heuristics do not guarantee optimality, thus affect accuracy, but can be easily scaled and provide solutions for big data sets.

In DES, model size is defined only partially in terms of the number of objects considered (number of processor units, number of workers, etc.). The model structure considers the types of objects used and, more importantly, the number and type of relationships among the objects. Size, and thus complexity, is heavily dependent upon structure and in particular the number and type of relationships. Even if a DES model has been designed to be easily scalable, the relationships among objects makes scalability in most cases a significant challenge — it is necessary to change the structure in order to change the size. In contrast to the DMPO models, the complexity of a DES model does not affect the choice of solution method and only slightly impacts solution speed (the model run time increases linearly with the problem size).

Typically, models are verified, at least initially, using small size and structure, typically few objects, few time periods, or both. However, it may be necessary to test models in larger contexts. For example, Case 1 was validated for a crossdock facility model with 3-input doors and 3-output doors, but a realistic case would be a 50-input doors by 50-output doors arrangement. Since it may be difficult to scale up the structure of simulation models, and since changing the optimization solution method requires considerable research and development, it is important to specify the size early on in the project.

OPERATIONAL DIFFERENCES

The second set of dissimilarities between DMPO and DES include operational differences in modeling the underlying system. These differences are described in terms of task dependencies, resource assignment and process logic.

Task dependencies

Precedence relationships are used to define the order in which tasks occur. For DMPO models, if the order is not a key consideration, it will typically not be included in the model for the sake of simplification and computation time. In that case only the number of tasks happening in a time interval will be considered; the order, the batch size, and the parallelism of the tasks are not taken into account.

However, in DES modeling, processing order is inherent: typically, unless explicitly specified, tasks are executed in first in, first out order. This fundamental difference can lead to discrepancies between the models during validation. For example, consider a single-channel process (c = 1) working at a rate \( r \) and a multi-channel process with \( c \) channels and rate \( r/c \) per channel. The output from the two options appear to be the same — they are on the average, but may not be true within a time interval. We use the pallet transfer process from Case 1 as an illustration. Assume the transfer rate per resource is \( r = 10 \) pallets/hour and the number of available resources is \( c = 3 \). If an outbound truck arrives at 10:00, then any pallet transferred from inbound before that time goes to storage, while any pallet processed after 10:00 goes directly into the outbound truck. A process with capacity \( c = 1 \) and rate per channel of \( r = 30 \) pallets/hour transfers each pallet in 2 minutes. Therefore, between 9:55 and 10:00, two pallets are processed and they both go into storage. However, a process with capacity \( c = 3 \) and rate per channel of \( r = 10 \) pallets/hour transfers each pallet in 6 minutes. Therefore, between 9:55 and 10:00, no pallet is fully transferred and no pallet goes into storage.

Modelers need to be aware of how basic processing order, batch size and precedence relationships are handled in each type of model. Typically, this is implicit in DES models and explicit, and often ignored, in DMPO models.

Resource assignment

The basic manner in which resources are selected for use may differ between DMPO models and DES models, thus leading to discrepancies in results and validation challenges. A common application of mathematical programming models is to make assignments between resources and tasks, as in our Case 2. When DES is used to test the implementation of an assignment, the default simulation logic may not result in comparable results. For example, in a simulation if a task needs to be performed by a resource and several resources are available, a default first-in, first-out criteria may not match the optimized assignment. Therefore, information on the DMPO assignments must be provided to the DES model so that the task can select the appropriate resource. In addition, if none of the available resources result in a match with the optimized assignment, then logic must be provided in the DES model in order to guide the task’s selection from the available resources; or, the task must wait until the appropriate resource is available.

Similarly, in a DES model if a resource becomes available and there are multiple tasks that need to be completed, a default first-in, first-out criteria may not match the optimized assignment. Therefore, as indicated above, information on the DMPO assignments must be provided to the simulation so the resource can select the appropriate task. In addition, if none of the tasks result in a match with the optimized assignment, then logic must be provided in the DES model in order to guide the resource’s selection of the available task; or, the resource must be made idle and wait until an appropriate task becomes available.
In short, modelers need to address ways to incorporate DMPO results into DES models, typically by modifying default resource assignment logic inherent in the simulation.

**Process logic**

By its nature, DES is greedy, i.e., it processes all items (pallets in our cases) that are scheduled in an event (instance in time), while DMPO models can transfer less pallets per time period if it improves the objective function in the optimization. In our Case 1, in order to force the simulation model to obtain a result similar to the optimization model, the amount of pallets that can flow through the model during each time period needs to be limited. This can be accomplished by directly using the output of the DMPO model as input to the DES, i.e., the capacity of the transfer process in the simulation model. Of course, this capacity will need to vary over time. It is interesting to note that this adjustment may make the simulation closer to reality. Therefore, the process logic issue can be mitigated by adding logic to the simulation model that provides flexible capacity over time to the activity.

Since DES is event-driven, priorities are often required in order to represent the appropriate behavior. For example, if both an inbound truck needs to be unloaded and an outbound truck needs to be loaded, which should an available resource service first? In Case 1 and Case 2 we include process logic in the DES model to push items from inbound trucks and pull resources from the arriving outbound truck. The pulling algorithm gives the priority to the outbound trucks; thus, we first seek to fill the outbound trucks that have to leave rather than emptying the inbound trucks. This logic is similar to what a manager would do. However, it may not agree with the optimal solution given by the DMPO model in all cases. Therefore, it is important to note that, when testing a DMPO model with a DES model, the former gives the optimal solution (when exact solution methods are used) while the latter does not. While the simulation can be driven towards a solution closer to the optimal, it cannot determine the optimal solution unless it embeds an optimization module (this is the case (a) in Figure 1, and beyond the scope of this paper). Using the optimal solution determined by the DMPO model as an input in the DES model is a good approach, but the simulation model needs to include decision logic for handling changes due to stochastic events.

**CONCLUSIONS**

DES and DMPO modeling take very different approaches to address operations problems. Of course this is due to fundamental differences in the way the two types of models are structured and solved. Even though quite different, simulation and optimization are often used in complementary roles to improve the decisions that result from using the models. These inherent differences provide challenges to modelers, especially in validation and verification.

This paper is based on the cases of two simulation models that are used to evaluate the performance of DMPO models (integer programming, mixed integer linear programming) in a stochastic environment. Those application cases model cross docking problems, but the issues we point out can occur regardless of the application field. We describe several key challenges occurring when the simulation models have to be validated, i.e., when the behavior of the simulation model and the optimization model are compared under deterministic conditions. We offer suggestions for mitigating those challenges.

We hope that the insights given on these issues can and will encourage an increase in the use of DES to assess the performance of DMPO models. We also hope that other modelers encountering various modeling issues will be encouraged to communicate them so that the community can benefit from their experience.

**REFERENCES**


Function Slices: A Model to Extract Parallelism from Sequential Applications

Nils Potthoff, Carsten Gremzow, Christoph Brandau, Dietmar Tutsch
University of Wuppertal – Automation / Computer Science
email: {potthoff, gremzow, brandau, tutsch}@uni-wuppertal.de

KEYWORDS
program transformation, Computer software model, software engineering, multiprocessor, control and data flow, automatic parallelization

ABSTRACT
Software development is still largely based on the sequential programming paradigm. This however does not fit modern multi- or many-core architectures. Writing efficient code for parallel systems often costs more than the benefit it provides. A different solution is necessary to accomplish the balancing act between current processors and keeping existing source code and well-known development processes. One approach is the automatic generation of parallelized executables from a sequential functional description. This paper presents a model and strategy to generate fragments of the original source code for concurrent execution. The internal structure is therefore mapped to a novel representation. Characteristics of this model and their influence on finding and using parallelism in sequential software are shown. The fundamental units in this model have known and fixed relations among each other. This will allow a rearrangement without side effects. To implement automatic slicing, information regarding the inter-procedural dependencies, extracted from an interprocedural data flow analysis is necessary. A static analysis cannot provide sufficient information about all data dependencies. When it is insufficient, runtime analysis will be used. Based on the control and data flow analysis parallelizable regions can be found. Different strategies for restructuring the software are possible. The transformations are not based on the parallel programming paradigm, but completely renounce the explicit programming of parallelism as is possible with programming interfaces such as MPI or OpenMP. Furthermore, other limitations in the sequential code like nested loops cannot be assumed. The analysis is independent from a special target architecture. The interconnection to code generation tools can be used for application synthesis.

Introduction
Multicore architectures are widespread and get more and more important in all kinds of devices. The number of cores will rise in the next years. Future architectures may have heterogeneous structure, with cores that vary in type. A higher performance with reduced power consumption can be reached. The software development must therefore be adapted to benefit from the advantages these architectures provides. The most important programming languages do not bring sufficient support for concurrency. Development processes that are proven and familiar are still driven by the sequential programming paradigm.
To achieve efficient programming of simultaneous processes there is a need for knowledge regarding the architecture and parallel algorithms. It is often necessary to have explicit programming of the synchronization and coordination of the single parts. Not only does software development reach a higher level of complexity, it also increases the potential for mistakes. In the development process one has to consider the algorithmic aspect on one hand. On the other hand hardware influences the software development, too. Multiprocessor systems or multicore processors are connected through a network or via shared memory. There are two adequate and common programming interfaces: MPI (Message Passing Interface; (Pacheco 1996)) on one hand, and OpenMP (Open Multi-Processing; (Chandra et al. 2001)) on the other. OpenMP is an example for communication via shared memory.
The development of concurrent executable algorithms can rely on different types of parallelism, depending on the granularity. Parallelism on instruction level refers to a processor being able to execute several operations at once. Compilers can optimize code on instruction level at compile time. Parallelism on the data level as well as on the task level can use several cores.
Even if the application was ported to a special architecture, it is not designed for a changing number of cores. The parallelism cannot be scaled due to the fixed number of tasks that the program consist of. Data parallelism does not only depend on the specific task but also on the size of the input data set. In contrast to parallelism on the instruction level, which can be realized with compiler technologies, other forms of parallelism must be programmed explicitly. Semi-automatic techniques for the parallelization of systems with distributed memory are based on inscriptions for data fragmentation encoded by the developer (Callahan and Kennedy 1988).
A further aspect of the fragmentation mentioned above
is the effort to create and manage threads in addition to the exchange of data. Processors with more threads than cores would lead to further losses of performance. To generate efficient programs with multiple concurrent working parts without additional work in development, an automatic parallelization could be a solution. In this regard, it is very important to have knowledge about relations, especially the flow of control and data (Potthoff et al. 2015). In order to determine important information such as loop bounds or pointer accesses, it is not enough to rely on known compiler techniques. It is known that a static analysis cannot be sufficient. To allow every freedom in the design of sequential code, a dynamic analysis is unavoidable.

To reach a general applicability it is important to give all freedom in the sequential source code. The possibilities a programming language provides should also not be restricted. Additionally, it arises from the target architecture evolution mentioned before not to be bound to a special target architecture. The analysis and automatic transformation of instructions should be independent of an architecture. The mapping of components to cores can follow.

**State of the art**

Many numerical, mathematical or signal processing algorithms can be represented by nested loops. Concepts of loop parallelization can be used for automatic transformation in this case. In particular this applies to the Polytope Model (Lengauer 1993). The absence of subprograms in nested loop programs (subprograms are normally used for casing or structuring or respectively for avoiding repetition of code) makes it possible to have easy static predictability of data and control flow dependencies.

The **SUIF-Project** (Amarasinghe et al. 1993) is based on the transformation of loops for shared memory architectures. **SUIF²** executes an interprocedural analysis (Aigner et al. 2000). Regarding the limitations in the original source code, the method presented in (Li et al. 2012) is limited to scalar data. The **S2P Tool** also uses the static analysis of the source code to generate task or loop parallelism. Only a dynamic analysis of data flow is able to determine dependencies comprehensively. Decoupled Software Pipelining (DSWP) (Ottoni et al. 2005) extracts pipeline structures from the source code, but no other type of parallelism. The selection of a loop structure from an NLP-Description for parallelization is made by **Helix**. The source code is supplemented in so far as dependencies sustained (Campanoni et al. 2012). A further method to analyze loops on a source code level and to extract pipeline stages is presented in (Cordes et al. 2011). This is based on integer linear programming and uses a combined control and data flow graph.

Important information regarding possible parallelism is derived from runtime behavior by **pMapper**. Furthermore, it provides special algorithms that handle the processing. Parallelism is thus not derived from analysis (Bliss 2007).

In the past different fragmentation models like superblocks (Hwu et al. 1993), Hyperblocks (Mahlke et al. 1992) and Treegons (Banerji et al. 1997) were proposed to make usage of instruction level parallelism. These units do not have the characteristic used for our parallelization approach. The rearrangement of superblocks by Ye and Chen (2012) is also based on runtime analysis. Compared to the approach presented in this paper, a superblock is more coarse granular. Moreover, not only the complete data flow, but also some extracted characteristics are used here. An interprocedural shape analysis and its application for parallelizing programs with dynamic data structures were presented in (Asenjo et al. 2008). **CECUS**, a compiler infrastructure, uses a different approach to analyze data dependencies, namely consulting tests on data independency (Dave et al. 2009).

A concept based on dynamic analysis was proposed by Tournavitis and Franke (2010), in this work machine based learning is used. Vanderendouck et al. (2010) make use of dynamic profiling, too. They presented a parallelization based on annotations by the developer. A tool can help with suggestions.

**Code Fragmentation**

With the sequential programming paradigm the compiler translates the source code to a list of instructions which fits the single processor architecture. The match of description, executable and architecture is obvious. In modern architectures the abstraction level differs. There should be tasks assigned to operation units like cores or processors. Every task then consists of a set of instructions. Hence, a program description needs to be translated into these coarse grained tasks. With scheduling of tasks additional dependencies must be taken into consideration. A set of instructions of any size can be put in execution when it is known, how many and which data should be processed. The producer of this data must be known and the results must exist and must be available completely. Every task thus has static data flow dependencies. Otherwise, the correct execution order cannot be guaranteed. It follows that an automatic parallelization must first translate the sequential source code into units with static data dependencies. A function does not fulfill this constraint. From the source code it is obvious where a function is called but a function can enclose branches. The processed data within the surveyed function is not static in either case.

A further known structure are basic blocks. A basic block has only one entry point and ends at one point with a jump instruction (branch, switch, or return). The consumed data is fixed, but the source or producer of them is not. Pointers and pointer arithmetic allows to
alter data sources and control flow sources. In order to determine important information such as loop bounds or pointer accesses, it is not enough to rely on known compiler techniques. For example a function

```c
fun(char *data, int length){...}
```

can operate on data arrays with unknown length. The length is determined at runtime and passed to the function as second parameter. A subroutine call can modify pointers just as well; hence every call possibly changes the dependencies. It is conceivable that a basic block operates on one object (variable) but in fact reads or modifies different objects underlying. Instruction are executed several times. If the modified data was referenced by a pointer, the target can change in between. The source code indicates one object, at execution different memory areas are read or modified. A static code analysis cannot provide this information (Hind 2001). The solution to get units with postulated static data dependencies is to sub-divide basic blocks into subblocks at call instructions. A part of a basic block produced like this is called subblock (SB). A basic block BB featuring N call instructions is thus subdivided into N + 1 subblocks. Figure 1a shows a hypothetical basic block featuring one call and the resulting subblocks SB0 and SB1. The shortest sequence of instructions between alternatively

- a basic block label
- a call instruction

and alternatively

- a call instruction
- a branch instruction
- a switch instruction
- a return instruction

is now called block B.

Compilers usually decompose programs into their basic blocks. Subdividing basic blocks into subblocks lead to a control flow graph like shown in figure 3. Even a less complex application collapses into a multiplicity of subblocks. For further operations a reduction of complexity would be preferred. The characteristics which led to this structure must not be changed. It is noticeable that

```c
struct stuff {
  int cnt;
  char buf[1024];
};
volatile void foo (char *buf, int len_foo, char fac_foo) {
  int i;
  for (i = 0; i < len_foo; i++)
    buf[i] = fac_foo;
}
volatile void bar (char *buf_bar, int len_bar, char off_bar) {
  int i;
  for (i = 0; i < len_bar; i++)
    buf_bar[i] = off_bar;
}
volatile void foobar (char *buf_foobar, int len_foo, char fac_foo) {
  foo (buf_foobar, len_foo, fac_foo);
  bar (buf_foobar, len_foo, fac_foo);
}
int main (int ac, char **av) {
  int fd;
  struct stuff st;
  char fac, off;
  (fd = open (av[1], O_RDONLY));
  fac = (char [atoi (av[2])];
  off = (char [atoi (av[3])];
  while ((st.cnt = read (fd, st.buf, 1024)) > 0) {
    foobar (st.buf, st.cnt - 24, fac, off);
    write (2, st.buf, st.cnt);
  }
  close (fd);
}
```

Figure 2: Example application for linear data transformation – Source code.

there are sequences of blocks within one function. If a sequence is not interrupted by another function, the dependencies could not be changed externally. This means that no unpredictable modifications of data or pointers can take place. For example BB0-SB2 and BB1 of function foobar in figure 3 is such a sequence of two blocks. The required static data dependencies are still met if these blocks were combined to a new unit. This combination is called function slice (FS) and offers a coarse grained and less complex model for further handling. A function slice is now defined as a set of blocks B. The combination is to maximize the unit of contiguous executable instructions without violating the condition of
3. $FS_n$ and $FS_m$ arise from different functions. The data dependence then can only exist through pointer referenced objects.

In case of 1a) and 2b) the objects are on the stack including potential function parameters. In case of 1b), 2c) and 3) objects referenced belong to the stack of a different function call or they are global objects on the heap.

The dependencies in 1a), 2a) and 2b) can be determined by a static data flow analysis. Static and dynamic analyses thus provide comprehensive information about control and data flow. It is clear, so far, that obviously there are no static predictable data dependencies. How to investigate them within a runtime environment is shown in the next section.

A further step to reduce complexity of application can be function slice clusters ($FSC$). The start sequence $(main)\to(open)\to(main)\to(atoi)\to(main)\to(atoi)$ is executed only once and as one block with no interruptions. The absence of back directed control flow edges indicates that this sequence can not be entered at any other point. No external modifications can occur. This sequence can be clustered and handled as a unit. Correspondingly, every sequence framed by back

---

**Figure 3:** Example application for linear data transformation (see fig. 2) – Control flow on Subblock level

---

**Figure 4:** Example application for linear data transformation (see fig. 2) – Function slice graph
Nodes: function slices, Solid arrows: control flow dependencies, Dashed arrows: data flow dependencies

---

constant data dependencies. The characteristic of basic blocks to have only one entry and only one exit point must also apply to function slices. Basic blocks arising from a hypothetical function and their control flow relations are shown in Figure 1 b) - d). Grouping the basic blocks and basic block sequences results in three function slices. Every stringent sequence of blocks composes a function slice. The instructions of a slice lie between a label at the beginning and a branch, switch, return or call instruction at their end, too. Figure 4 shows the example function slice graph of the example application. The possibilities of visualization were presented in (Potthoff et al. 2015).

Let a function slice $FS_m$ be dependent on $FS_n$ ($FS_n \to FS_m$). Investigating the possible data dependencies between these two function slices $FS_m$ and $FS_n$ the following combinations could occur:

1. $FS_n$ and $FS_m$ arise from one function $F$ and there is no $B_x$ being part of both $FS$ or $BS$ arised from $B_x$. The dependency then can only exists through
   (a) local function data objects relating to $F$
   (b) pointer referenced objects.

2. $FS_n$ and $FS_m$ arise from one function $F$ and there is one shared $B_x$ which is part of both $FS$ or from which at least two $BS$ emerged and both $FS$ contain at least one of these $BS$. The data dependence can then only exist through
   (a) $B_x$ local (function local) objects
   (b) function local data objects relating to $F$
   (c) pointer referenced objects.

---

125
directed control flow edges can form a cluster.

**Global Dataflow Analysis**

To get the intra-procedural and inter-procedural data flow in quality and quantity, the LLILIA Framework (Grenzow 2007) is used. This framework is based on the Low Level Virtual Machine (LLVM) which is a compiler infrastructure with a language independent instruction set and type system (Lattner and Adve 2004). The instruction set of the internal representation uses a static single assignment form (SSA). Intermediate code is well suited for machine-independent optimizations as it is not bound to an architecture. Furthermore, accessing the runtime environment is less complex in a virtual machine than in a physical target architecture.

To determine the control flow a static analysis is used. Exemplary figure 3 visualizes possible control flow. LLVM provides bit code with block structure. Every block begins with a label and ends with a jump instruction (branch, switch, return). A correlation of labels and jump instructions yields the control flow on block or function level.

For quantitative data flow tracing the combination of sources and corresponding sinks must be determined. Static analysis is not sufficient. To realize a complete analysis even for pointer referenced objects the source code is automatically augmented with profiling instructions by LLILIA. The enriched code is compiled and executed in the runtime environment (RTE). While running, profiling information is written. The runtime profile and its analysis is used for extraction of the data flow which is not determined statically. Store and following read instructions on the same memory area are captured. In detail every data consumer (block or function) obtains the calculation of the previous modifier. One block $B_n$ depends on block $B_m$ ($B_n \rightarrow B_m$) if both access one shared data object. This may be given via pointer referenced objects. If both blocks arise from one basic block, the dependency can additionally exist through basic block local data objects. If both blocks are part of one function, local function objects including function parameters can result in a data dependency, too. Dynamic program analysis is unsafe and cannot guarantee completeness. We use statistical valuation of multiple runs with different input data sets combined with coverage analysis. This reduces the risk of missing dependencies in the resulting flow graph. The partitioned program representation can also be the starting point for code generation.

**Parallelization strategies**

The function slice model can be used to identify and use parallelism beyond the instruction level. Every function slice contains a fragment of the original program description. This unit encapsulates some functionality as a set of instructions. The dependencies of function slices are fully known and unchangeable after the analysis. Most important are the data dependencies, as they are elementary for the functionality of the program. Control flow dependencies are eligible to find loops in the program. Parts of a software program can be executed concurrently when there is no data dependency between them. A function slice contains some operations. All data consumed by these instructions must be available at execution time. The constraint for the execution schedule is to compute and store required data before. When and where the operations are executed, does not matter.

Different operations of rearranging of function slices can be performed. The aim is to find concurrent executable regions and reach the maximum level of parallelism at runtime.

**Elimination of unrequired control flow edges**

When investigating FS-graphs, it becomes apparent that a significant number of function slices has no data dependencies, they just pass the control flow. In the example application from figure 2 the function foobar calls subfunctions and assigns data objects. The function itself contains no data modifying instructions. In the function slice representation these slices can be elim-
Figure 6: Comparison of maximum control flow graph depth of the MPEG2Decode application in function slice view without considering interprocedural dataflow dependencies (black graph) and with function slice parallelization (gray graph). Overall application acceleration is 3.5 fold.

...inated. Adapting the control flow pointing from the predecessor to the successor does not change any functionality. After execution of the last data relevant function slice, the next slice with data dependencies can be executed instead of the formerly next slice (in sequential control flow sequence).

A second characteristic to be used is that there is a limited number of data flow edges. In strictly directed control flow graphs the elimination of control flow edges leads to purely data flow graph. All important dependencies are preserved and the maximal possible parallelism arises. Figure 5a) shows the combined data and control flow graph of the original program. In figure 5b) the resulting data flow graph shows the possible parallelism. Function slices without any effect to the functionality of the software system become apparent (gray nodes). These blocks do not need to be executed. The maximum depth of the graph in a) is significant larger than in b). The degree of parallelism is obvious. A further aspect is that in this example more than one start point of the application exists. It has not to start with one thread and spread later but can run with concurrent threads from beginning on.

Loop operations

If the control flow graph contains back directed edges, it means that there are loops in the software system. Parallelizing loops has to consider mainly two characteristics:

1. For the data dependency between loop cycles following situations can exist:
   (a) A loop pass depends on a previous one
   (b) All loop cycles are data independent

2. The count of loop cycles can be
   (a) fixed
   (b) calculated before entering the loop for the first time
   (c) changed through operations in the loop body

In case 1. and 2. possibility parallelization can be performed through pipelining. To realize pipelining the loop body has to be divided into stages. An algorithm to find pipeline stages can operate on the combined control and data flow graph. Back directed control flow edges mark loop areas and determine the required margin. Every loop has to be regarded as a potential candidate for program transformation. The division of the loop body is driven by data dependencies. Vertices without data edges can be eliminated first. Data structures that pass through the pipeline should be processed in every stage to achieve the optimal communication-to-operation ratio. The analysis separates data being processed from parameters, by the data volume handed over. The control and data flow graph allows automatic analysis and the determination of pipeline stages.

In case 2.a) unrolling can be calculated at compile time. Unrolling or loop splitting is possible in case 2.b), too. In this case the breaking it into multiple loops which have the same bodies but iterate over different contiguous portions of the index range must take place at runtime.

Results

To test and evaluate the methods presented above different applications were used. Their complexity varied from simple test constructs to complex benchmark applications.

Figure 2 shows the source code of an example application with a pointer referenced data object passed to subroutines. The extracted function slice graph with data and control flow edges is shown in figure 4. Figures 7 and 8 show the results of runtime analysis. The chronological behavior of this application is visualized. In figure 7 the active slices and the passed control flow edges are plotted. The time represents the execution time between steps. The loop is obvious as periodic structure. In figure 8 the appendant data flow is plotted. Function Slices with no data dependency were removed. Here the data source and sink of a flow and the lifetime of information can be read. Dependencies between different loop cycles would get visible (see figure 9).

The analysis of data flow through runtime analysis and the determined data volume of all data flow edges is shown in figure 10. The analysis of possible pipeline parallelism was used for the example application (see figure 4). Resulting pipeline stages are marked in figure 4. The output is in line with the expectations.
To verify the functionality of the described process analyzing the MPEG2 decode and encode reference implementation of the MPEG Software Simulation Group was shown. The call graph of the MPEG2 decode implementation contains 621 vertices, the subdivision into function slices results 1471 vertices. Figure 6 shows the distribution of function slices per control flow graph depth in the original FS-graph compared to the distribution with inherent parallelization by exploiting function slice data independencies only. No further parallelization techniques such as loop unrolling, pipelining or speculative execution of function slices have been employed. For the resulting graph the depth of every vertex was calculated. The quantity of vertices with the same depth is plotted over the depth. It is evident that the resulting graph is less deep but has more breadth thus multiple data independent function slices can be executed in parallel on different cores. For the example of the MPEG2Decode application the overall application acceleration is 3.5 fold without applying further optimizations such as loop unrolling etc.

**Conclusion**

Modern processors with multi or many core architectures facilitate concurrency. Multithreading support is necessary to use this capability within an application. A sequential program description must be divided on thread-level. In this process the functionality must not be changed. Manual parallelization has many disadvantages. Automatic parallelization can avoid these drawbacks.

Sequential software is being restructured. This process is not based on the parallel programming paradigm, but completely renounce the explicit programming of parallelism.

A novel model that allows extracting elementary parts of software was developed. This approach has several advantages. The concept presented is characterized by high flexibility. The initial source code has no limitations relating to the programming language and its features like nested loop programs (NLP). As LLVM supports various front ends the tool chain will not only be usable for C-Code but also for any language supported by LLVM. Furthermore, the introduced method is independent from a special architecture as it uses a virtual architecture.

**REFERENCES**


DEVS
Optimizing distributed DEVS simulations with partitioning and Hidden Markov Model learning methods

Christopher Herbez & Eric Ramat
LISIC, ULCO
50 rue Ferdinand Buisson, Calais, France
email: [herbez,ramat]@lisic.univ-littoral.fr

Gauthier Quesnel
MIAT, INRA
24 chemin de Borde Rouge, Castanet-Tolosan, France
email: gauthier.quesnel@toulouse.inra.fr

KEYWORDS
Simulation, DEVS, Graph weighting, Partition, Hidden Markov Models

ABSTRACT
With the emergence of parallel computational infrastructures at low cost, reducing simulation time becomes again an issue of the research community in modeling and simulation. In this context, our previous papers presented a method to reduce the simulation time in a parallel DEVS context. This approach reduces simulation time without reaching the maximum gain. The partitioning method used does not take into account the dynamic of models. To address this problem, we propose in this paper an approach to weight the model graph to take into account this dynamic when partitioning. This paper presents the weighting graph process by learning of the dynamic of models states using Hidden Markov Models. The purpose of this article is to determine the quality of this weighting method by a comparison with a simulation approach.

INTRODUCTION
Modeling and analysis of complex systems dynamics are becoming increasingly costly in time and memory capacity. The multi-modeling is a response to the increase demand for coupling heterogeneous models. Obviously, this process leads to the increase in computation demand and therefore, the increase of computation time. It is therefore important to think about the good use of new physical processor infrastructure (multicore, multiprocessor and grid). Work in this area is not new: includes all work around distributed simulation [Chandy and Misra, 1979, Chandy and Misra, 1981] but also work on parallel computing [Fujimoto, 1990]. However, our main concern is the construction of an optimized organization of simulators as part of DEVS (Discrete Event Specification). The DEVS formalism [Zeigler et al., 2000] and Parallel-DEVS variant [Chow and Zeigler, 1994] is a candidate to develop a response at the same time formal and technical. That is a modeling tool of discrete events and simulation theory with a hierarchical approach. The global model, called structure of the model in DEVS terminology, is a graph of coupled models. Our approach is to flatten the hierarchy in order to obtain the graph of models. This graph is partitioned in order to parallelize the models execution as efficiently as possible.

In [Herbez et al., 2015a], we presented this approach as well as the relative gains obtained for two types of partitioning. One is based on the connectivity of the graph, and the other is oriented modeler. In these examples, the gain obtained by the introduction of a good partitioning is about 20% compared to an initial model hierarchy. Thereafter, in [Herbez et al., 2015b], we showed the limits of this approach. Currently, the dynamic of models is not taken into account when graph partitioning, which explains the limited gain observed. In this paper we propose an approach to weight this graph in order to better reflect the dynamic of models. The first part describes the Parallel-DEVS formalism and our approach. The second part present the methodology implemented to weight the model graph using the Hidden Markov Models. Finally, the last part evaluates the quality of the graph weighting obtained by our approach and allows to validate its.

EFFICIENT DISTRIBUTION OF MODELS IN A PARALLEL DEVS CONTEXT
Our research is mainly devoted to the optimization of discrete simulations in a parallel DEVS context. This paper presents the changes that we propose in order to optimize these simulations in time and memory. For this, we propose a modification of the DEVS hierarchical structure. This section presents the DEVS formalism and the approach set up to change the hierarchy of models.

Parallel-DEVS context
DEVS [Zeigler et al., 2000] is a high level formalism based on the discrete events for the modeling of complex discrete and continuous systems. The model is a network of interconnections between atomic and coupled models. These models are in interaction via time-stamped events exchanges.
More specifically, we present the Parallel-DEVS (PDEVS) formalism [Chow and Zeigler, 1994, Chow, 1996]. This extension of the classic DEVS introduces the concept of simultaneity of events essentially by allowing bags of inputs to the external transition function. Bags can collect inputs that are built at the same date, and process their effects in future bags.

PDEVS defines an atomic model as a set of input and output ports and a set of state transition functions:

\[ M = \langle X, Y, S, \delta_{\text{ent}}, \delta_{\text{ext}}, \delta_{\text{con}}, \lambda, ta \rangle \]

With: \( X, Y, S \) are respectively the set of input values, output values and sequential states

\( ta : S \to \mathbb{R}_0^+ \) is the time advance function

\( \delta_{\text{ent}} : S \to S \) is the internal transition function

\( \delta_{\text{ext}} : Q \times X^b \to S \) is the external transition function

where:

\[ Q = \{ (s, e) | s \in S, 0 \leq e \leq ta(s) \} \]

\( Q \) is the set of total states,

\( e \) is the time elapsed since last transition

\( X^b \) is a set of bags over elements in \( X \)

\( \delta_{\text{con}} : S \times X^b \to S \) is the confluent transition

function, subject to \( \delta_{\text{con}}(s, \emptyset) = \delta_{\text{ent}}(s) \)

\( \lambda : S \to Y \) is the output function

If no external event occurs, the system will stay in state \( s \) for \( ta(s) \) time. When \( e = ta(s) \), the system changes to the state \( \delta_{\text{ent}} \). If an external event, of value \( x \), occurs when the system is in the state \((s, e)\), the system changes its state by calling \( \delta_{\text{ext}}(s, e, x) \). If it occurs when \( e = ta(s) \), the system changes its state by calling \( \delta_{\text{con}}(s, x) \).

Every atomic model can be coupled with one or several other atomic models to build a coupled model. This operation can be repeated to form a hierarchy of coupled models. A coupled model is defined by:

\[ N = \langle X, Y, D, \{ M_d \}, \{ I_d \}, \{ Z_{i,d} \} \rangle \]

Where \( X \) and \( Y \) are input and output ports, \( D \) the set of models and:

\( \forall d \in D, M_d \) is a PDEVS model

\( \forall d \in D \cup \{ N \}, I_d \) is the influencer set of \( d \):

\( I_d \subseteq D \cup \{ N \}, d \notin I_d, \forall d \in D \cup \{ N \}, \)

\( \forall i \in I_d, Z_{i,d} \) is a function,

the i-to-d output translation:

\( Z_{i,d} : X \to X_d, \text{ if } i = N \)

\( Z_{i,d} : Y_i \to Y_d, \text{ if } d = N \)

\( Z_{i,d} : Y_i \to X_d, \text{ if } i \neq N \) and \( d \neq N \)

The influencer set of \( d \) is the set of models that interact with \( d \) and \( Z_{i,d} \) specifies the types of relations between models \( i \) and \( d \).

The atomic and coupled models are respectively associated with simulators and coordinators. The aim of simulators is to compute the various functions while the coordinators manage the synchronization of exchanges between simulators (or coordinators in a hierarchical view).

**Optimization of the DEVS hierarchical structure**

Many DEVS implementations reduce the hierarchy to only one level: a coordinator and several simulators. [Muzy and Nutaro, 2005] introduce this type of structure called flattening. The first step of our approach is to flatten the hierarchy in order to obtain a graph of model. This work is possible thanks to the closure under coupling property of DEVS [Zeigler et al., 2000]. This property describes formally the coupled model is equivalent to an atomic model. Thus, an atomic model can be moved into a new coupled model and all the hierarchy of coupled model can be merged into a unique coupled model. The connections between atomic models of this coupled model give an oriented graph called graph of model. The Figure 1 give an example of flattening of the hierarchy.

![Hierarchical model structure](image)

![Flat model structure](image)

**Figure 1**: At the top, an example of hierarchical structure of a DEVS model. At the bottom, new structure after flattening of the hierarchy.

Our approach is centered around the restructuring of the hierarchy so that it is optimal for distributed simulation. The optimality depends on two main factors:

- charge balance between different parties to execute a maximum of models in parallel
- minimizing of exchanges between different parts to avoid increasing the transfer costs related to information exchange between hosts
The restructuring is done by a partitioning of the models graph. It is made in order to comply with the conditions presented above.

The partitioning phase of our approach is presented in details in our previous paper [Herbez et al., 2015a]. It consist to apply the multilevel method presented in [Karypis and Kumar, 1998]. This method is divided into 3 phases:

- Coarsening: Graph reduction by successive vertices matching, while keeping the nature of the original graph. The Heavy Edge Matching introduced in [Karypis and Kumar, 1998] is implemented for this phase.

- Partitioning: Creating of a partition \( P_k \) of the coarsening graph \( G_n \) using a partitioning heuristic. We choose an expanding region method: the Greedy Graph Growing Partitioning presented in [Bichot and Siarry, 2013].

- Uncoarsening: Projection of the partition \( P_k \) on each contraction graph levels with a refinement for keep a good quality. We use a local optimization algorithm based on Kernighan-Lin algorithm [Kernighan and Lin, 1970].

This method generates a partition of atomic models minimizing the optimality criteria for distributed simulation. From this partition, a new two-level hierarchy is built. This hierarchy consists of the same number of coupled models that there are parts in the partition. The atomic models of the same part are assigned to the same coupled model. And these new coupled models are distributed on the network of machines available for the simulation. The Figure 2 illustrates the construction of the new hierarchy from a partitioning on the previous example.

![Models graph partitioned](image)

![Optimised hierarchical model structure](image)

Figure 2: At the top, example of partitioning of the models graph. At the bottom, new hierarchical structure at 2 levels obtained from this partition.

The current problem is that the model graph as it is constructed does not contain information regarding the information transmission frequency from one model to another or on their execution time. However, these information are crucial for the partitioning quality because they correspond to the weighting of the graph and reflect the dynamic of models. For now, this weighting is set to 1 by lack of information, which damages the partitioning quality. To address this problem, the following section presents an approach to weight the graph by learning of model dynamics.

**Weighting of model graph**

The model graph is an oriented graph where each vertex corresponds to an atomic model and each edge at information flow linking two atomic models. Graph theory, presented in [Bichot and Siarry, 2013], authorizes the weighting of these elements. For a vertex, weight is a quantification of the running time of the model that it symbolizes. When a model has a longer execution time, the associated vertex will have a greater weight. For an edge, the weight quantifies for one hand the amount of data exchanged between two models and for the other hand their frequencies. The aim of our approach is to develop a graph weighting system without knowledge of the internal dynamic of models.

Individually, each atomic model is considered like a black box because the internal dynamics of the model is assumed unknown. The unique information provided by models are the duration \( d_i \) which are the elapsed time between each emission of external events via the output function \( \lambda \). Moreover, models inform us if they are in a "Infinite state" or not. We call "Infinite state", a state whose the duration is infinite. This assumption is strong but essential for the effective convergence of learning algorithms.

From these informations, a learning algorithm is performed to create a probabilistic states automaton. These automatas must be able to generate sequences of observations statistically close to those of the model. The advantage of this approach is to generate a very large number of observations without being penalized by the calculation times that require multiple executions of a model. This is possible provided that the learning model is less expensive than the generation of observation sequences by simulation.

The observation sequences obtained by simulation of probabilistic states automaton allow to estimate the emission frequency of outputs by calculating a weighted mean. Consider an observation sequence of emission duration between two outputs consisting of \( k \) distinct values \( d_1, \ldots, d_k \) and denote \( n_i \forall i = 1, \ldots, k \) the number of times where the duration \( d_i \) appears in the sequence. The average duration of emission between two outputs

135
is given by:
\[
d_{\text{mean}} = \frac{\sum_{i=1}^{k} d_i \times n_i}{\sum_{i=1}^{k} n_i}
\]

Its provides the average frequency of issue:
\[
f_{\text{mean}} = \begin{cases} 
1 & \text{if } f_{\text{mean}} \to 0 \text{ slow model} \\
> 1 & \text{fast model}
\end{cases}
\]

The weighting of the edges is given by emission frequency average \( f_{\text{mean}} \). When generating the observation sequence by simulation, the elapsed time to obtain for a set of \( N \) durations is recorded. With this information, we can weight the edges of the graph by this time.

We start from the assumption that our simulation consists of \( K \) types of different models. The goal is to achieve a learning for each type of model in order to establish a weighting rule. This approach allows us to minimize the number of learning steps. The following section outlines the learning method used to model the dynamics of the models.

**LEARNING OF ATOMIC MODELS DYNAMIC USING HIDDEN MARKOV MODELS**

Knowing that, for each model, the set \( S \) of states is unknown and that only a series of observations \( Y = \{y_1, \cdots, y_k\} \) is available through simulation, we choose to model the dynamics of models from a Hidden Markov Model (HMM) introduced by Baum and his colleagues in [Baum et al., 1970]. The following section presents the hidden Markov models.

**Generalities on Hidden Markov Models**

A Markov model is a sequence of random variables \( (S_n, \ n \in N) \) which allows to model the dynamics of a system. The feature of this process is that the current state \( S_n \) is independent of the past state \( S_{n-1} \), which is expressed mathematically by:
\[
P(S_n = j | S_1 = i_1, \cdots, S_{n-1} = i_{n-1}) = P(S_n = j | S_{n-1} = i_{n-1})
\]

Having no knowledge of the dynamics of the model, it is necessary to assume that every state is reachable from any system status. Moreover, each transition is associated at an observation which is the duration of emission \( d_i \) between two outputs. Considering this information, it appears that an ergodic Hidden Markov Model is a good candidate to model our states graph. One of the features of a HMM is a finite symbol alphabet is associated with each state. In our case, symbols correspond to period \( d_i \) observed by simulation.

The HMMs are defined in [Rabiner, 1989] by five parameters:

- \( N \) the number of states in the model. Where the states are defined by \( S = \{S_1, \cdots, S_N\} \).
- \( M \) the number of symbols. Where symbols are defined by \( V = \{v_1, \cdots, v_M\} \).
- \( A = \{a_{i,j} | i, j = 1, \cdots, N\} \) the state transition probability distribution.
- \( B = \{b_{i,j} | i = 1, \cdots, N \ ; j = 1, \cdots, M\} \) the observation symbol probability distribution. Where \( b_{i,j} \) is the probability that the state \( S_i \) emit the symbol \( v_j \).
- \( \pi = \{\pi_1, \cdots, \pi_N\} \) the initial state distribution.

For clarity, we call an HMM by \( HMM = \{\pi, A, B\} \), where \( \pi, A \) and \( B \) are built such that:
\[
\sum_{i=1}^{N} \pi_i = 1, \sum_{j=1}^{N} a_{i,j} = 1 \text{ and } \sum_{j=1}^{M} b_{i,j} = 1
\]

The Figure 3 illustrates an ergodic Hidden Markov Model by an example.

![Figure 3: Example of ergodic HMM with 3 states and 3 symbols](image)

For each learning, an HMM is created with a view to answering the following problem: *Starting from a set of observations \( O = \{O_1, \cdots, O_k\} \), how to adjust settings of an HMM to model best the process?*

The following section presents the algorithms used for the HMM learning.

**Learning Methods of HMMs**

Learning determines the best HMM parameters to represent observations \( O \) ie, which attribute to \( O \) the best
The probability of occurrence. We denote $P_{HMM}(O)$ the probability of occurrence of the sequence $O$ assigned by the HMM, the goal is to estimate the parameters $HMM$ that maximizes $P_{HMM}(O)$:

$$HMM = \arg \max_{HMM} P_{HMM}(O)$$

The learning process begins with system initialization $HMM = \{\pi, A, B\}$. As we have no knowledge except the observation sequence $O = \{O_1, \ldots, O_k\}$, it is necessary to begin by arbitrarily fix the number of system states. For the number of symbol, it suffices to determine the different number of observations in the sequence (assuming that the sequence contains all the symbols). The sets $\pi$, $A$ and $B$ can be randomly constructed (provided they respect the properties of an HMM) or with equal probability distributions. It is very difficult if not impossible to get the perfect $HMM$. However, there is a method to get a good approximation: the Baum-Welch algorithm presented in section 4 [Bilmes, 1998]. The Baum-Welch algorithm is a learning algorithm derived from the EM algorithm (Expectation Maximization). Given a set of observation sequences $O$ and an initial model $HMM$, the Baum-Welch algorithm undertakes a re-estimation of the parameters $(\pi, A, B)$ of the model so as to increase generating probabilities of these observation sequences. To optimize computation steps, the Baum-Welch algorithm uses the Forward and Backward algorithms also detailed in [Bilmes, 1998]. The procedure is to re-estimate the parameters of the model and to recalculate the likelihood of the new model. The process is re-iterated until the current maximum likelihood.

The following section presents the relationship between DEVS models and HMMs.

**Building of learning sets from atomic DEVS model simulations**

The atomic models of which we seek to model the dynamics are classified into two categories:

- models without input : they are simply represented by a conventional HMM (see Figure 3).
- models with inputs: a "modified" HMM is used to be able to represent the states called "Infinite state".

Indeed, it is essential to represent the states where the model is waiting for an input event. For this we add the symbol $\infty$ corresponds to a state of infinite duration. This additional symbol will also lead us to modify the generation algorithm by simulation.

On the other hand, the learning of models with inputs requires the creation of a generator. This generator is intended to simulate potential model inputs. To ensure the best exploration of the dynamics of the model, the generator must explore the widest possible spectrum of the input values $(d_i)$. For simplicity, we consider that the events do not carry data.

The generation of learning sets from individual atomic models, produces sequences of type $(d_i)$ where $d_i$ is either a positive integer value or infinity for "Infinite states". There is one sequence by model type. This sequence is then divided into sub-sequences. Each sub-sequence has a random length and ends with an end symbol. This splitting is random. It is assumed that the splitting will not lead to bias in learning. From these sub-sequences, HMMs are constructed by learning. Figure 4 illustrates the learning process.

![Image]

**Figure 4**: Learning scheme of the weighting of the model graph

The following section presents how the HMM obtained by learning allows to create the weighting of edges of the model graph.

**Building weighted graph from HMM models**

The learning of models is performed using one or more sequences of observations obtained by simulation. The sequences should be neither too large in order to avoid spending too much computation date nor too short not to lose information on symbols. From these observations, we first construct the set of symbols $V$ and then initializes the Hidden Markov Model $(\pi, A, B)$. For models with inputs, we add the symbol $\infty$ to the set $V$. The number of states is fixed at $N = 6$ and the distribution probabilities are almost equal for $A$ and for $B$. Once the $HMM$ constructed from the Baum-Welch algorithm, the complete graph is reconstructed by replacing the atomic models by their $HMM$. We simulate the automaton graph at probabilistic states. This simulation is based on the observation sequences generation algorithm for models without entry (see Algorithm 1) and on a variant of the previous algorithm for models with inputs see Algorithm 2).
The random selection for a state or an observation, according to the probabilities \(\pi\), \(A\) and \(B\), requires the creation of sets of cumulative probabilities \(\pi_c\), \(A_c\) and \(B_c\). Where

\[
\pi_{c_i} = \sum_{k<i} \pi_k + \pi_i
\]

\[
A_{c_{ij}} = \sum_{k<j} A_{ik} + A_{ij}
\]

\[
B_{c_{ij}} = \sum_{k<j} B_{ik} + B_{ij}
\]

To simulate this selection, we select three values \(k_\pi, k_a, k_b \in [0, 1]\) with a uniform distribution. To determine the initial state, we search \(j\) such as \(k_\pi \in [\pi_{c_{j-1}}, \pi_{c_j}]\). Similarly, to determine the future state and the emitted symbol, we search \(j\) such as \(k_a \in [A_{c_{ij-1}}, A_{c_{ij}}]\) and \(k_b \in [B_{c_{ij-1}}, B_{c_{ij}}]\).

**Algorithm 1 Observation sequence generation for model without input**

1. procedure
2. select an initial state \(S_i\) with probabilities \(\pi\)
3. set initial state \(S \leftarrow S_i\)
4. select a symbol \(v_k\) with probabilities \(b_{ik}\)
5. seq \(\leftarrow \{v_k\}\)
6. while \(S \neq S_N\) do
7. select a future state \(S_j\) with probabilities \(a_{ij}\)
8. \(S \leftarrow S_j\)
9. select a symbol \(v_k\) with probabilities \(b_{jk}\)
10. seq \(\leftarrow\) seq \(\cup\) \(\{v_k\}\)
11. return seq

Changes that we must achieve in the simulation algorithm are twofold: to take into account the symbols \(\infty\) and interaction between the HMMs. In algorithm 1, the sequence may be entirely generated because it is not dependent on external information. However, in the case of models with inputs, we must take account of events produced by the upstream models of the simulated model. The sequences \(\{d_i\}\) are converted into events. Events are placed in input of the models and the arrival dates are defined by the \(d_i\). Events are considered in the simulation of HMMs when the automaton sends the symbol \(\infty\). Indeed, when the symbol \(\infty\) is generated, the automaton waits for an external event (as defined in the DEVS formalism). However, if the transmitted symbol is not \(\infty\) and an external event happens then the event is ignored. So unlike the conventional case, the HMM simulator is piloted by the presence of the symbol \(\infty\) and the arrival of events. Finally, emission time average is determined between two outputs on all edges of the graph. This weighted average is the computation of the expectation of the variable of symbols derived of the symbol \(\infty\): \(V - \{\infty\} = \{v_1, \ldots, v_{M-1}\}\). At each \(v_j\) is associated a probability \(P(v_j) = \sum_{i=1}^{N} b_{ij}\) which is the sum of the emission probabilities of the symbol \(j\) for each state \(S_i\). The expectation of the variable \(V\) is:

\[
E(V) = \sum_{j=1}^{M-1} v_j P(v_j) = \sum_{j=1}^{M-1} v_j \sum_{i=1}^{N} b_{ij} = \sum_{j=1}^{M-1} \sum_{i=1}^{N} v_j b_{ij}
\]

The entire process of weight computation will be illustrated in an example of model graph presented in section Model and graph.

**DATA PRESENTATION AND RESULTS**

To illustrate the process, we will use an example based on a graph of atomic models. This graph consists of a single type of deterministic model. For each model, the parameters are different which implies different dynamics from the point of view of \(d_i\). Using this example, we will show the impact of the approximation due to HMM. It seems obvious that the building by learning of HMM leads to a loss of accuracy compared to the original models. We will quantify this loss and show that the impact on the estimated weight is low.

**Models and graph**

Our tests are made from only one type of models. An atomic model represents the filling of \(N\) tanks of same
size, where each has its own filling speed \( s_i \). The dynamic is this: when \( K \) tanks have reached the maximum capacity \( Q_{\text{max}} \), they are emptied. If all tanks are full so an output event is generated. At this time, a \( d_i \) is calculated. Figure 5 illustrates the operation of this model. Depending on the setting, one sees a dynamic that can be represented by a finite state automaton. The \( d_i \) are shown on the automaton edges and vertices represent states where the tanks are all full.

![Tank model example and state graph](image)

**Figure 5:** Tank model example and state graph

Then, we introduce an external event at the deterministic model. The tanks are under the influence of upstream tanks. When tanks were full \( k \) times, downstream tanks fill faster. We model this mechanism by: when \( k \) external events arrived, the filling speed is increased for one of the tanks. The speed will be reset to the initial state when all speeds have been increased.

![Atomic model graph with \( d_{\text{mean}} \)](image)

**Figure 6:** On the left, atomic model graph with \( d_{\text{mean}} \). On the right, HMM graph with \( d_{\text{mean}} \)

### Results

The first step of the method allows us to get the \( d_i \) for all atomic models. The models \( A, B \) and \( C \) have same values for parameters. In this way, a single simulation is required (see figure 7). In the general case, it is important to identify identical models to minimize the number of simulations. For the other five models, three simulations are needed: \( D \) and \( E \) are identical and \( H \) have no exit. These three simulations require the setting of input generators: each generator produces \( d_i \) randomly according to a uniform law whose boundaries are between the minimum and maximum of \( d_i \) observed at the output of the upstream models. We have four sets of \( d_i \):

\[
\{15; 3; 15; 1; 2; 12; 1; 17; 2; 4; 9; 10; 15; 1; 2; 2; 3; 8; 7; 8; 4; 11; 3; 12; 3; 12; 1; 2; 1; 11; 6; 9; 15; 3; 14; 3; 9; 3; 15; 3; 1; 11; 3; 15; 3; 2; 13; 1; 2; 12; 6; 13; 2; 3; 12; 15; 3; 1; 5\}; \{9; 3; \ldots \}; \ldots
\]

**Figure 7:** \( d_i \) of A model with stable sub-sequence

According to the method, the \( d_i \) are subdivided to obtain a set of observation sub-sequences for the learning step. We took the opportunity to compute the duration average \( d_{\text{mean}} \) (see figure 6).

Two parameters are to be set for the learning step (Baum-Welsh algorithm): the number of maximum of iterations and the number of hidden states. In our case, we fix them respectively 100 and 10. The number of hidden states leads to highly variable quality results. It is important to fix them an adequate way. By example, one would be to vary them to retain the optimal number. Some methods exist (for example, [?]).

![HMM graph with weights](image)

**Figure 8:** HMM graph with weights

Then, from HMMs constructed by learning, an HMMs graph is built and simulated. We then obtain the simulated \( d_{\text{mean}} \). Figure 6 shows the differences between the simulated \( d_i \) from DEVS models and simulated \( d_i \) from HMMs. They are different but the results are close. To improve these results, it is necessary to introduce a method for optimizing the number of hidden states, to study the impact of the initial conditions of learning and also to study the setting of generators for models with
inputs. These three elements determine the quality of HMMs. In addition, the DEVS models treated in the example are deterministic and show sequences that are repeated in the $d_i$. What is the impact? What happens if there is no repetition or if the sequences are too short? The last step is to convert the $d_i$ to weight (see figure 8) and make partitions from the flattening graph [Herbe et al., 2015b]. Partitioning is not addressed in this paper.

CONCLUSION AND PERSPECTIVES

The optimization of the hierarchical structure of the DEVS simulation is not limited to a simple partitioning of the model graph. As we have shown in our previous work, the partitioning process does not take into account the dynamics of the models, which is reflected by a lower gain in performance. In this article, we proposed a method of weighting of the model graph with Hidden Markov models. As we have seen in the results section, several factors may impact the quality of learning. It is therefore important to study the different impacts. In addition, the DEVS models that were used to validate the approach, belong to a subclass of DEVS models (deterministic and finite states). We must therefore expand the types of models (eg, stochastic). Similarly, $d_i$ are integer and their number is finite. It is important to extend our results to models whose $d_i$ are real and whose number is infinite. The definition of $d_i$ classes could be a solution. There are also two other restrictions: the models have only one input port and one output port, and events do not carry data. The first restriction is not complicated to take into account, in particular, in the simulation algorithms of HMM. However, it is more difficult to take into account data. We should build specific generators for models with inputs.

One last important perspective is the study of the transformation function of $d_i$ in weight. This function is now quite simple and non-linear. Another way, by example linear, might be a better choice.

REFERENCES


A NEW APPROACH TO MODELING DYNAMIC STRUCTURE SYSTEMS

Oumar Maïga\textsuperscript{1,3}  
\textsuperscript{1}Université des Sciences, des Techniques et des Technologies 
Bamako, Mali 
maigabababa78@yahoo.fr

Hamzat Olanrewaju Aliyu\textsuperscript{2,3}  
\textsuperscript{2}School of Info. & Comm. Tech. 
Federal University of Technology 
Minna, Nigeria 
hamzat.aliyu@futminna.edu.ng

Mamadou Kaba Traore\textsuperscript{3}  
\textsuperscript{3}LIOMS CNRS UMR 6158 
Université Blaise Pascal 
Clermont-Ferrand, France 
traore@isima.fr

KEYWORDS
Dynamic Structure Systems, DEVS, DSDEVS, HiLLS

ABSTRACT
This paper presents a novel approach to modeling Dynamic Structure Systems (DSS) using the High Level Language for System Specification (HiLLS), a graphicontextual language that combines system-theoretic and software engineering concepts from DEVS (Discrete Events System Specifications) and Object-Z respectively, to form a unified language for specifying Discrete Events Systems for multiple analysis methodologies. We take benefit of the expressive and user-friendly notations of HiLLS to alleviate the complexity of modeling dynamic-structured systems and use Dynamic Structure DEVS (DSDEVS) as one of its semantics domains to take advantage of the latter’s simulation protocol and its associated tool(s). We provide as a case study, the modeling of a Cash Deposit Machine of an automated teller machine to illustrate how HiLLS is used to model DSS and its equivalent DSDEVS model for simulation.

INTRODUCTION
We present a simple approach to the Modeling and Simulation (M&S) of structural changes in complex Discrete Events Systems (DES) using a graphical modeling language. A DSS is a system whose structural properties, such as number of components and/or connections/interactions between them may change dynamically at runtime. Examples of such systems exist in communication networks where connections are dynamically established and broken between nodes, automatic switching systems where a component may be linked to different peer components under different conditions, etc.

In M&S, modeling structural changes in a system is usually an herculean task as most modeling tools lack the required constructs to easily express such properties. With DEVS (Zeigler et al. 2000) as starting point, a number of formalisms have been defined to provide sound theoretical background for the simulation of DSSs; notably among them are DSDEVS (Barros 1995), dynDEVS (Uhrmacher 2001) and rhoDEVS (Uhrmacher et al. 2006). They however lack easy-to-use concrete syntaxes for defining models independently from mathematical equations or program codes; we complement these abstract formalisms with a user-friendly front end. We propose a graphical modeling language, HiLLS, that supports the specification of a broad range of complex systems including DSSs. HiLLS combines system-theoretic and software engineering concepts from DEVS and Object-Z (Smith 2000) respectively to define an integrated vocabulary of systems constructs for analyses within disparate semantic contexts. We will however limit our discussions about HiLLS in this paper to the M&S of DSSs. This is achieved by adopting DSDEVS as a semantics domain for HiLLS. We prefer DSDEVS among its contemporaries for its safe and simple approach to maintaining consistency between systems’ structure and behavior (Muzy and Zeigler 2014).

We present overviews of DEVS-based formalisms for modeling DSSs in the next section, followed by the abstract and concrete syntaxes of HiLLS as well as its mapping to DSDEVS (for simulation semantic). Then we illustrate the approach with a case study before presenting our conclusions and perspectives in the last section.

STATE OF THE ART
DSSs are characterized by operations that manipulate the set of components, couplings, interfaces and behavior of a system (Hu et al. 2005). We present overviews of existing formalisms for modeling DSSs. For a better understanding of this section, we assume the reader has a basic understanding of the DEVS formalism; a brief introduction is however provided in the appendix for interested reader.

DSDEVS is a variant of DEVS with capability for modeling structural changes. It retains the specification of atomic DEVS while introducing a "network executive" model into the coupled network specification to manage structural changes in the latter. The state variables of network executive store the structural information of the network so that there is a one-to-one correspondence between the executive’s instantaneous state and the network’s structure.

DSDEVS coupled network is described as: 
\[ \text{DSDN} = (\chi, M_{\chi}) \]  
where \( \chi \) is the name of the network
executive and $M_x$ is the dynamics of network.

$$M_x = (X_x, S_x, s_{a,x}, Y_x, \gamma, \Sigma^*, \delta_x, \lambda_x, \tau_x)$$

$X_x$ and $Y_x$ are the input and output interfaces of the system, $S_x$ is the set of states, $s_{a,x}$ is the initial state, $\Sigma^*$ is the set of possible structures of the network, $\gamma : S_x \rightarrow \Sigma^*$ is the function which associates a unique network structure to each state of the executive, $\forall \Sigma \in \Sigma^* \land \sigma_2 \in S_x : \gamma(s_\Sigma) = (D, \{M_i\}_{i \in D}, \{I_i\}_{i \in D}, \{Z_i\}_{i \in D})$

$\delta_x : \Sigma^* \times (X \cup \emptyset) \rightarrow S_x$ is the transition function

$\lambda_x : S_x \rightarrow Y$ is the output function and

$\tau_x : S_x \rightarrow R^+ \cup \{+\infty\}$ is the time advance function.

In DSDEVS, only single central network executive is responsible for the management of structural changes in the network. Any other component is not allowed to modify the network structure or its own structure and/or behavior. This prevents ambiguity when different components require structural change. The single executive also ensures structural and behavioral consistencies.

dynDEVS (Uhrmacher 2001) and rho-DEVS (Uhrmacher et al. 2006, Muzy and Zeigler 2014) are other DSS modeling approaches based on the DEVS formalism. Unlike DSDEVS, these formalisms allow naturally dynamic behavior at atomic level and dynamic structure at network level. In addition, rho-DEVS supports modeling changes in components interfaces. (Muzy and Zeigler 2014) proposed a more general and decentralized approach to modeling DSS in which each component can change its own behavior and network structure.

While these formalisms provide the formal background for M&S of DSS, model specification can be very laborious because modeller must list all the possible structures of the system as a part of the state space of the network executive which can grow exponentially depending on the number of components and dynamic couplings between them. Like DEVS itself, they don’t have concrete syntaxes and logical semantics.

HILLS proposes a simpler approach to modeling structural changes without need for a separate "executive" model or special transition functions to capture structural information: these information are inherent in the system’s configuration and appropriate structural changes occur naturally with state transitions. The ability to specify this aspect in a graphical language also makes our approach easier to use and accessible to a larger audience.

**HILLS**

HILLS evolves from the DEVS-Driven Modeling Language (DDML) (Traoré 2009, Maiga et al. 2012, Ighoroje et al. 2012), a graphical modeling language built on DEVS to facilitate the use of the latter by domain experts via user-friendly graphical concrete syntax to describe system models. The goal of HILLS is to be able to create multi-semantic models that can be used for simulation, formal analysis and enactment.

HILLS’ syntax combines system-theoretic and Software Engineering concepts adopted from the DEVS and Object-Z respectively. Our choices of system constructs from Object-Z and DEVS are motivated by their universalities in their respective domains; while the former claims suitability for modeling most kinds of state-based systems for formal analysis, the latter has been proven to be a common denominator to most DES simulation formalisms (Vangheluwe 2000). Another advantage of the combination is that Object-Z provides constructs such as predicates and expressions that are reused for the refinement of abstract constructs such as states and transitions functions adopted from DEVS. This feature also aids the synthesis of executable program codes for enactment.

In addition to the DEVS-based system-theoretic concepts in HILLS, the syntax also adds concepts to describe structural changes in DSSs. HILLS’ approach to modeling DSSs is unique in that it provides a simple and graphical means of doing it; we demonstrate this with a case study in a later section.

**Abstract Syntax**

Figure 1 is an excerpt of the HILLS’ abstract syntax. In the bottom-right segment of the diagram (within the dashed-box), a DES is described as an $HSystem$ which may be an atomic unit or composed of interacting components ($hcomponents$). It may have input and/or output ports (class $Port$) for interacting with its environment by means of exchanging messages called $Events$. By its inheriting the class $HClassifier$, an $HSystem$ may have a $StateSchema$ in which state variables are declared with possible constraints, an $AxiomaticSchema$ that defines global parameters, and $Operations$ that manipulate the system’s variables and parameters.

The behavior of an $HSystem$ is described by a finite set of configurations and transitions between them. A $Configuration$ is a cluster of all states satisfying a unique system property defined on the state variables. The $sojournTime$ of a configuration is the duration for which it may be assumed before a scheduled transition occurs. A configuration is regarded as $Passive$, $Transient$ or $Finite$ if its $sojournTime$ is positive infinity $(+\infty)$, zero $(0)$ or positive real number greater than zero $(\mathbb{R}^+)$ respectively. A $ConfigurationTransition$ belongs to one of three categories: an $internal$ transition occurs at the expiration of the $sojournTime$ of the current configuration, an $external$ transition occurs whenever an $input$ is received before the end of the $sojournTime$ while a $confluent$ transition occurs when the reception of an input coincides with the expiration of the $sojournTime$. A transition is accompanied by a sequence of $computations$ that manipulate the state variables to satisfy the property of the target configuration; it may also involve
sending events to some output ports.

Coupling describes the relations between the ports of the components of a system. Systems influence on one another exchanging events (i.e., messages, impulse, etc.) through their input and output ports. Therefore, a coupling is a property that establishes a relation between a source port (sender) and a target port (receiver) for the exchange of events.

In DSSs (e.g., automatic switching systems), certain couplings are characteristics of some states of the system, hence coupling and decoupling of components occur during state transitions. Specifications of structural changes follow a similar fashion in HiLLS; a couplings is a kind of property that defines a configuration. Therefore, the couplings associated with the configurations of an HSystem naturally define the instantaneous relationships between its components; this is in fact similar to the real-time behavior of such systems. InputCoupling, InternalCoupling and OutputCoupling, have the same definitions as DEVs’ EIC, IC and EOC respectively.

In addition to the amenability of Object-Z to formal analysis, the level of functional refinement provided by the segment of the meta-model outside the dashed-box helps to precisely and completely model systems’ behavior in a generic form that can be refined to executable program code for the enactment of systems. These concepts are reused for the refinement of the system-theoretic concepts through their associations with them; examples are the associations between the following pairs of components: (Property, Predicate), (Port, Declaration) and (Event, Expression).

Concrete Syntax

The concrete notations to express HiLLS’ concepts are described in Figure 2(a-h). HClass (a) is denoted by a box with three compartments similar to the UML class symbol. The first compartment contains the HClass’ name and parameters if any. The second compartment houses the state and axiomatic schema if any. We adopt the notations of the state schema and axiomatic schema as used in Object-Z. The third and last compartment houses the definitions of the class’ operations if any. An operation is similar to the state schema but with additional information indicated on its top side. The top bears the name attribute of the operation, the list of parameter declarations (if any) and the type of the operation. Similarly, an empty type bracket denotes an operation that does not produce any output. Associations such as inherited classes, composition, and class reference use the corresponding notations in as used in the UML class diagram notations.

The HSystem (b) notation extends that of HClass; it has four compartments with the first three serving similar functions as in HClass while the fourth contains the transition diagram that describes the system’s behavior. The input and output interfaces are denoted by windows attached to the left and right sides respectively of the second compartment. In each rectangular window, a port is denoted by a small arrowhead labelled with the port declaration.

The notation for a finite configuration (d) is a box with five compartments for label, properties, sojournTime,
activities, and sub-configurations respectively from top to bottom. Passive configuration (c) is similar to a finite configuration except that the compartment for sojourn Time is not represented; a vertical stripe is attached to its right side as an indication of its infinite sojourn Time. Transient configuration (e) is denoted by a circle with three compartments for its label, properties and activities if any. Its shape depicts its zero sojourn Time.

Configuration transitions are represented by arrow-ended lines emanating from source to target configurations with associated computations as textual labels. A conditional statement in the path of a transition may initiate a choice between one of two targets. In such cases, the condition is enclosed within a diamond (h). To disambiguate the flow of the computations, the transition arrow flows into the left corner of the diamond and flows out from the circle attached to the right corner if the condition is true; otherwise, it flows out from either the top or bottom corner.

Semantic Mapping of HiLSS to DSDEVS

HiLSS has a family of semantics domains, each element providing a context for system analysis; however the focus of this paper is limited to DSDEVS as a semantics domain for the simulation of DSS. By generating an equivalent DSDEVS model from a HiLSS specification, we take benefit of the simulation protocol of the former and its associate tools. Due to space limitations, a detailed formal specification of the semantic mapping cannot be given in this paper but the description provided in this section will help the reader follow subsequent sections.

DEVS, being a mathematical formalism solely for system specification has no specific constructs for representing objects. Since the formalism also does not prescribe any concrete syntax, the user may take advantage of the freedom to represent an object as a mathematical structure with its essential attributes and operations constituting the elements of the structure. HiLSS' operations are also specified as mathematical functions that may be called from the DEVS-specific functions such as the transition and output functions.

An HSystem maps to Atomic or Coupled DSDEVS models if its hComponents is empty or not respectively. In the case that it maps to a Coupled DSDEVS, the set HSystem.hComponents.target maps to the set of its components. HiLSS' ports map to the set of corresponding input and output ports of the equivalent DSDEVS model. If an HSystem maps to an Atomic DSDEVS, each configuration, c, translates to a subset of the state set S of the latter while the sojourn time, \( f_c \), of c translates to the time advance function, \( t_a(s) \), of the states in this set.

Internal, External and Confluent transitions in HiLSS are extracted to build DSDEVS \( \delta_{int} \), \( \delta_{ext} \) and \( \delta_{conf} \) functions respectively. The output computations accompanying internal and confluent transitions in HiLSS are used to build the \( \lambda \) function in the DSDEVS model. In the case that an HSystem maps to a coupled DSDEVS, the configurations and transitions of the former map to the states and state transitions of the network executive of the latter.

**CASE STUDY: THE CASH DEPOSIT MACHINE**

The Cash Deposit Machine (CDM) allows a customer to deposit a bundle of currency notes into an account. It comprises six components that collaborate to process the currency bills. It first checks the genuineness of the bills by testing for some security properties, recognised bills are returned to the customer user while accepted bills are temporarily held in the machine to request a confirmation of the transaction from the user. If confirmed, the bills are permanently stored in the machine while the transaction runs to completion. Otherwise, the bills are returned while the transaction is being cancelled. The following are the components and their respective roles in processing the bills.

1. **Bundle Acceptor (BA)** receives a bundle of bills (max. 50 per transaction), releases them into the BC one at a time for validation and notifies the controller after sending the last bill. It also receives returned bills from the REJ and presents them in a bundle to the customer in the event of recognised bills and/or cancellation of transaction. It is guarded by a shutter that opens only when bills are taken from or returned to the user.

2. **Bill checker (BC)** receives a bill at a time from the BA to test its genuineness. Accepted and rejected bills are passed on to the ES and REJ respectively.

3. **Escrow (ES)** is a temporary storage for valid bills (max. 50) until the transaction is confirmed or cancelled. If confirmed, the bills are sent to the CAS; otherwise, they are released into the REJ. ES has only one output slot that may be linked to either the REJ or the CAS depending on the situation.

4. **Reject box (REJ)** stacks rejected or cancelled bills and returns them in a bundle to the BA for onward delivery to the customer upon receiving the appropriate control instruction.

5. **Cassette (CAS)** is a permanent storage for deposited bills.

6. **Control Board (CON)** is a micro-electronic board that coordinates the activities by sending instruction signals to other components.

**Note:** This case study has a static set of components but the couplings between certain components vary dynamically. The ES has one output port that may be
coupled to CAS or REJ depending on whether the user confirms or cancels the transaction.

**HiLLS Model of the Case Study**

The HiLLS model in Figure 3 shows the hierarchical compositions between the CDM and its autonomous components that interact with one another by exchanging bills and low voltage signals. Due to space limitation, we will present the detailed specification of one of the components, REJ and that of the CDM to show how dynamic couplings between some components are specified.

**Bill Model**

Since it has no autonomic behavior, the bank note (Bill) is modelled as an HClass as shown in Figure 4. It has four parameters d, t, l and w representing the denomination, thickness, length and width respectively of the bank note. The state schema declares the state variables and constraints that define the features of the bill.

**REJ Model**

The REJ model is shown in Figure 4. The input interface has two ports i1 and i2 for receiving bills and digital signals respectively. Similarly, the output interface has two ports o1 and o2 for sending bundles of bills and digital signals respectively.

*State Variables and Constants: REJ declares variables interrupt of type positive integer, current of type Bill and stack which is a list of Bills. By default, every instance of HSystem declares an implicit variable, duration of type positive real number (with possibility of positive infinity) to hold the instantaneous values of the sojourn times of active configurations. The axiomatic schema in the second compartment defines two constants: limits and period representing the component’s capacity and time taken to exchange bills respectively as specified in the system’s synopsis. Note that limit is used as the upper limit of the cardinality of the state variable stack (see the composition relation ‘stack’ from REJ to Bill indicating the maximum number of bills that may be stacked in the REJ at any instant and for any transaction. The Init operation initializes the global constants with the parameters provided at instantiation. Operations: The third compartment defines three operations; getStackLenght, getPeriod and load. In HiLLS, any operation with type different from void has an intrinsic variable, out, of the same type as the operation’s type; this variable must be assigned the computed value of the output of the operation. It serves the same purpose as return in Java.

*Configurations* The fourth and last compartment contains the specification of REJ’s behavior described by configurations and transitions. The state space is partitioned into four configurations waiting, loading, offloading and reporting with their respective properties indicated in their second compartments. As stated previously, the computed sojourn time for each configuration is assigned to the implicit variable duration. loading and offloading, being finite configurations, have their sojourn times explicitly defined in the third compartments while waiting and reporting have implicit sojourn times of positive infinity $+\infty$ and 0 respectively.

*Configuration Transitions: With waiting as the initial configuration, there are seven (7) transitions as shown in the diagram. The triggers of external and confluent transitions are represented by the port and event names at the source end; e.g., an external transition from wait-
ing → loading is triggered by \([i?x](i.e., \text{when input } x \text{ is received on port } i)\). Output events that may accompany internal and confluent transitions are represented as assignment of values to appropriate port variables in the transition computations. For example, during the transitions offloading → reporting and reporting → waiting, o1stack and \(o2\)0 are outputs on ports \(o1\) and \(o2\) respectively.

**CDM Model**

We have seen in Figure 3 that the CDM has six components: \(ba, bc, es, rej, cas\) and \(con\) which are instances of \(BA, BC, ES, REJ, CAS\) and \(CON\) respectively. Figure 5 shows the internal details of CDM’s specification. In addition to the six components, CDM declares two state variables, \(transaction\) and \(shutter\) that keep tracks of the presence of an active transaction and the status of the shutter respectively. The shutter is the opening through which bills are presented to or taken from the machine. \(delay\) is a parameter of the CDM that describes the maximum period for which the shutter remains opened when accepting bills from the user. The second line of predicates in the state schema specifies the invariants for static couplings between certain components of the system. These invariants are realized by the initialization of corresponding port variables in the \(connect\) operation. For example; a coupling specification \(A.port_r = B.port_s\) implies that port \(s\) of system \(B\) influences port \(r\) of system \(A\). In addition to initializing the system’s parameter \(delay\), the \(Init\) operation of CDM also invokes the \(Init\) of each of its components.

In addition to the explicit and implicit specifications of sojourn times for configurations demonstrated in the \(REJ\) specification, the CDM presents two special functions, \(\eta\) and \(\pi\), to specify the sojourn times in some specific cases. \(\eta\) is used in cases where a configuration has some sub-configurations where its own sojourn time cannot be precisely specified. Hence, the function indicates that the sojourn time at any instant is equal to that of its active sub-configuration at that time. For example, the sojourn time of processing at any instant is equal to that of whichever is active among validation, confirmation and escrow-dispense. \(\pi\) on the other hand is used when all sub-configurations under the same parent have identical sojourn time; the sojourn time may be specified once on the parent configuration while all its children inherit this property from it. For instance, configurations completing and cancelling have identical sojourn times which is specified once on escrow-dispense from which all its sub-configurations inherit the property. Configuration transitions of the CDM can be read just the same way those of \(REJ\) were presented previously.

**DSDEVS Equivalent Model of the CDM**

We present in this sub-section, the DSDEVS model of the CDM derived from the Hillis specification.

\[
DSD_{CDM} = (X, M_X) \text{ where } X = CMD \text{ and } M_X = (X_X, S_X, s_{o_X}, Y_X, \gamma, \Sigma^*, \delta_X, \lambda_X, \tau_X)
\]
\[ X_\chi = \{(i_1, \{0, 1\}), (i_2, \{0, 1\}), (i_2, PBill)\} \]
\[ Y_\chi = \{(o_1, PBill), (o_2, \{0, 1\})\} \]
\[ S_\chi = \{(\text{transaction}, Z^+), (\text{shutter, String}), (\text{conf, CONF})\} \]

where: \(\text{CONF} = \{\text{idle, accepting, returning, validating, confirmation, completing, cancelling}\} \)

\[
\delta_{oX} = \begin{cases} 
(\text{transaction} = 0, \text{shutter} = \text{"closed"}, \text{conf} = \text{idle}) 
\end{cases}
\]

\[ \Sigma^* = \{M_{\text{conf}}\}_{\text{conf} \in \text{CONF}} \text{ with } \]
\[ M_{\text{conf}} = \{D_{\text{conf}}, \{M_1\}_{i \in D_{\text{conf}}}, \{I_i\}_{i \in D_{\text{conf}}}, \{Z_i\}_{i \in D_{\text{conf}}}\} \]

Note that out of the three state variables, the system’s structure depends on the value of conf. i.e.,

\[ \Sigma^* = \{M_{\text{idle}}, M_{\text{accepting}}, M_{\text{returning}}, M_{\text{validating}}, M_{\text{cancelling}}, M_{\text{completing}}, M_{\text{confirmation}}\} \]

Let \( D_0 = \{M_{\text{name}}\}_{M \in \text{Comps}} \) where Comps = \{ba, bc, es, rej, cas, con\}

\[
\forall M_{\text{conf}} \in \Sigma^*, D_{\text{conf}} = D_0 \land \{M_i\}_{i \in D_{\text{conf}}} = \text{Comps} 
\]

\[ \forall s_2 \in S_\chi, \gamma(s_2) = M_{\text{conf}} \]

Let \( s_{\text{CDM}} = \{\}, s_{\text{ba}} = \{\text{rej}\}, s_{\text{bc}} = \{ba\}, s_{\text{es}} = \{bc, con\}, s_{\text{rej}} = \{con\}, s_{\text{cas}} = \{ba, rej, es\}, s_{\text{cas}} = \{\} \]

Recall that \( M_{\text{conf}} \) is the structure of the system for the actual value of the \(\text{conf}\) variable; due to space constraint, we present the details of \(M_{\text{completing}}\) and others can be obtained in similar manner.

\[ M_{\text{completing}} = \{D_{\text{completing}}, \text{Comps}_{\text{completing}}, I_{\text{completing}}, Z_{\text{completing}}\} \]

where:

\[ I_{\text{completing}} = \{s_{\text{CDM}} \cup \{ba\}, s_{\text{bc}}, s_{\text{cas}} \cup \{es\}, s_{\text{es}}, s_{\text{ab}} \cup \{CDM\}, s_{\text{rej}} \cup \{bc\}, s_{\text{con}} \cup \{CDM\}\} \]

\[ Z_{\text{completing}} = \{Y_{ba} \rightarrow Y_{\text{CDM}}, Y_{\text{completing, ba}} : X_{\text{CDM}} \times Y_{\text{ba}} \rightarrow X_{\text{ba}}, Y_{\text{completing, bc}} : Y_{\text{ba}} \rightarrow X_{\text{bc}}\} \]

\[ Z_{\text{completing, es}} : Y_{\text{bc}} \rightarrow X_{\text{es}}, Y_{\text{completing, rej}} : Y_{\text{bc}} \times Y_{\text{con}} \rightarrow X_{\text{rej}}\]

\[ Z_{\text{completing, cas}} : Y_{\text{es}} \rightarrow X_{\text{cas}}, Z_{\text{completing, con}} : X_{\text{CDM}} \times Y_{\text{ba}} \times Y_{\text{es}} \rightarrow X_{\text{con}} \]

\[ \text{Transition function: } \delta_\chi : Q_\chi \times (X_\chi \cup \phi) \rightarrow S_\chi \text{ with } \]

\[ \delta_\chi((0, \text{closed, idle}), e, i) = (1, \text{opened, accepting}), \forall e > 0. \]

This transition corresponds to the external transition between the idle configuration and the accepting configurations when an event is received on port \(i_1\). This transition creates also a new input coupling between the input port \(i_2\) of CDM and the input port \(i_1\) of its component \(ba\). The coupling creation is part of the changes specified in the set of different possible structures.

\[ \delta_\chi((1, \text{opened, accepting}), e, i) = (1, \text{closed, validation}), \forall i \in \{0, 1\}, e \geq 0. \]

This transition takes into account the two transitions (the external one and the confluent one) from the accepting configuration to validation configuration. This transition specifies a new internal coupling between the output port \(o_2\) of the \(bc\) and the input port \(i_1\) of the reject box \(rej\).

The external transition from confirmation when an input is received on port \(i_2\) is described as a piecewise function:

\[ \delta_\chi((1, \text{closed, confirmation}), e, i) = \begin{cases} 
(1, \text{closed, completing}) & \text{if } i = 1; \\
(1, \text{closed, cancelling}) & \text{if } i = 0; 
\end{cases} \]

The target of the transition is completing or cancelling if

The input received on port \(i_2\) is 1 or 0 respectively. The transition is accompanied by the establishment of couplings between the port \(o_2\) of \(es\) (i.e., \(es.0_2\)) and \(cas.i_1\) or \(rej.i_1\) if the value of the received input is 1 or 0 respectively.

The internal transition from the completing configuration to the idle configuration is represented as:

\[ \delta_\chi(1, \text{closed, completing}, 0, \phi) = (0, \text{closed, idle}). \]

Other transitions can be mapped in similar manner.

**Time advance function:** \(\tau_\chi : S_\chi \rightarrow R^+ \cup \{+\infty\}\)

\[ \forall s_\chi \in S_\chi, \tau_\chi(s_\chi) = \text{duration}(\text{conf}) \text{ i.e., } \]

\[ +\infty \text{ if } \text{conf} \in \{\text{idle, confirmation}\} \]

\[ t_{ba} \text{ if } \text{conf} = \text{validating} \]

\[ t_{es} \text{ if } \text{conf} \in \{\text{cancelling, completing}\} \]

\[ t_{rej} \text{ if } \text{conf} = \text{returning} \]

\[ delay \text{ if } \text{conf} = \text{accepting} \]

\[ t_{ba} = \text{ba.getPeriod()} * \text{(ba.getStackLength() + 1)} \]

\[ t_{es} = \text{es.getPeriod()} * \text{(es.getStackLength() + 1)} \]

\[ t_{rej} = \text{rej.getSpeed()} + \text{ba.getSpeed()} \]

\[ delay \text{ is a parameter of the CDM.} \]

**The output function:** \(\lambda_\chi : S_\chi \rightarrow Y_\chi \)

\[ \forall t \in \{0, 1\}, shut \in \{\text{opened, closed}\}, \]

\[ \lambda_\chi(t, 0, \text{shut, returning}) = 0 \]

\[ \lambda_\chi(t, 0, \text{shut, completing}) = 1 \]

\[ \lambda_\chi(t, 0, \text{shut, cancelling}) = \text{rej.getStack()} \]

**CONCLUSIONS AND PERSPECTIVES**

We have presented a new approach to modeling DSSs with HILLS. The advantage of the proposed approach is the possibility of simple and communicable graphical notations to describe structural changes in complex systems which are usually expressed either as mathematical equations or program codes by existing solutions.

HILLS models are amenable to analyses by simulation, formal methods and enactment but our discussions in this paper were limited to the provision of simulation semantics for DSSs using the DSDEVS as the semantics domain. This way, we take benefits of the ease and simplicity of HILLS for specification and the simulation protocols for DSSs defined by DSDEVS and possibly, its associated tool called DELTA Environment (Barros and Mendes 1997).

We presented a case study to illustrate how HILLS’ concepts and notations are used to achieve the graphical modeling of DSSs. It is however important to state here that our approach is limited to what is possible with DSDEVS as far as structural changes is concerned; for example, structural changes in the input and output interfaces are not supported. We are currently working on a HILLS editor that will allow for the specification of complex DESS and subsequent generation of simulation codes based on existing tools.
APPENDIX

Discrete Events System Specification (DEVS)

DEVS (Zeigler et al. 2000) is a system-theoretic mathematical formalism for specifying DESs as abstract mathematical objects for simulation. It supports the specification of a full range of DESs as other formalisms for systems in this category have been proven to have equivalent DEVS representations (Vangheluwe 2000).

Basically, DEVS defines two abstraction levels for DESs - atomic and coupled DEVS. An atomic DEVS has a time base; state, input and output sets; and functions that define successive states and outputs events. A coupled DEVS on the other hand is an hierarchical composition of two or more atomic and/or coupled DEVS as components with couplings between their input/output ports to enable their interactions.

Atomic DEVS, AM, is defined as:

\[ AM = (X,Y,S,\delta_{int}, \delta_{ext}, \delta_{conf}, \lambda, ta) \]

where:

\[ X = \{ (p,v) | p \in IPorts \land v \in dom(p) \} \]

\[ Y = \{ (q,v) | q \in OPorts \land v \in dom(q) \} \]

\[ S: \text{Abstract state set} \]

\[ \delta_{int}: S \to S \]

\[ \lambda \]

\[ \delta_{conf}: S \times X^b \to S \text{ with } Q = \{ (s,e) | s \in S \land 0 \leq e < ta(s) \} \]

\[ \delta_{conf}: S \times Y^b \to \lambda \]

\[ ta: S \to R_{+}^{\infty} \]

\( X \) and \( Y \) are the sets of input and output events respectively. \( IPorts \) is a set of input ports and \( OPorts \) is a set of output ports. An event, \( v \), in this context is a value generated in form of a message that triggers an action by its recipient. \( S \) is the set of states; at any given moment, the system is in a state \( s \in S \). The time advance function, \( ta \), maps each state to a specified duration after which a scheduled internal state transition, \( \delta_{int} \), is automatically fired. The external transition function, \( \delta_{ext} \), specifies the system’s response to the input event(s) before the expiration of the \( ta \) of current state; \( Q \) is called the set of total states and \( e \) is the elapsed time since the last state transition. If the input event coincides with the expiration of the \( ta \), then confluent transition function, \( \delta_{conf} \), is invoked instead.

The subscript \( b \) of \( X_b \) denotes a bag of input events. The function \( \lambda \) defines the outputs that may accompany internal and/or confluent state transitions. Similarly, the subscript \( b \) of \( Y_b \) denotes a bag of output events.

The Coupled DEVS, CM is defined as:

\[ CM = (X,Y,D,\{M_d\}_{d \in D}, EIC, EOC, IC) \]

where:

\[ EIC = \{ ((CM, iPCM), (d,ip_d)) | iPCM \in IPorts_{CM} \land ip_d \in IPorts_d \}_{d \in D} \]

\[ EOC = \{ ((d, op_d), (CM, op_{PCM})) | op_{PCM} \in OPorts_{CM} \land op_d \in OPorts_d \}_{d \in D} \]

\[ IC = \{ ((a, op_a), (b, ip_b)) | op_a \in OPorts_a \land ip_b \in IPorts_b \}_{a, b \in D} \]

\( X \) and \( Y \) are as defined for atomic DEVS and \( D \) is the set of names of components of \( CM \) such that \( M_d \) is the DEVS specification referred to by \( d \) for all \( d \in D \). An EIC, External Input Coupling, is a connection between an input port of \( CM \) and an input port of one of its components, an EOC, External Output Coupling, is a connection between an output port of \( CM \) and an output port of one of its components and an IC, Internal Coupling, is a connection between the output port of a component of \( CM \) and an input port of another peer component. More details on DEVS and its simulation protocols can be found in Zeigler et al. (2000).

REFERENCES


A FRAMEWORK FOR DISCRETE EVENTS SYSTEMS ENACTMENT

Hamzat Olanrewaju Aliyu1,2
1School of Info. and Comm. Tech.
Federal University of Technology Minna, Nigeria
hamzat.aliyu@futminna.edu.ng

Oumar Maïga2,3
2Université des Sciences, des Techniques et des Technologies Bamako, Mali
maigababa78@yahoo.fr

Mamadou Kaba Traore3
3LIMOS CNRS UMR 6158
Université Blaise Pascal Clermont-Ferrand, France
traore@isima.fr

KEYWORDS
Enactment, Discrete Events System, System Analysis, Prototyping, Design Patterns

ABSTRACT
This paper proposes a framework to guide the synthesis of program codes from Discrete Event Systems (DES) models for the enactment of the systems. Enactment in this context is the execution of a system's specification for real time verification of the specified properties and/or building a software solution for the system. Though it has been used extensively within the last decades to automate workflows and business processes, enactment is less pronounced mainstream computational system analysis domain. We believe that enactment of DES models can complement the more exploited methodologies like simulation and formal methods in model-based systems analysis. We propose a framework that provides a template for code synthesis from DEVS (Discrete Events System Specification)-based models and an execution protocol based on Object-Oriented Observer design pattern for the real time interpretation of system's properties. We provide a simple case study to illustrate the use of the framework.

INTRODUCTION
Simulation and testing have been used extensively within the last decades in the study, development and improvement of complex systems. Simulation and Formal Methods of various categories are most pronounced in this domain for studying dynamic and static system properties respectively. While simulation techniques are mostly scenario-based investigations of properties using time approximations through the advancement of execution time-at discrete steps-to the times of occurrences of events of interest, most formal methods deal with the static proof of the satisfaction or otherwise of certain quality and reliability properties (e.g., completeness, safety, deadlock freedom) throughout the entire life cycle of the system represented by the specification. It is a general belief that no single analysis methodology is sufficient to study all aspects of a system, hence multiple techniques are used to get complementary insights of the system.
Another system analysis (cum implementation) methodology which is more pronounced in business process management (BPM) (Van Der Aalst et al. 2003, Jeston and Nelis 2014) is enactment. In the field of BPM, enactment may be simply described as the execution of process definitions created by a workflow (Kouvas et al. 2010) where a workflow is described as the complete or partial automation of business processes during which a set of procedure rules is used to pass information and work lists from one participant to another for necessary actions (Otteinsooer and Fekete 2007). A more general software engineering description of enactment provided in (Dowson and Fernström 1994) is the execution or interpretation of software process definitions. According to the authors, an enactment mechanism may also interact with the environment (e.g., human-in-the-loop, software and hardware devices) to provide supports that are consistent with the process definitions; this property, interaction with external actors, is in fact another feature that differentiates enactment from mainstream simulation mechanisms in addition to the execution of system's functionalities in real time. Finally, in service engineering and Human-Computer Interaction, it can be inferred from (Holmäng and Evenson 2007) that enactment is used to describe the playing out of the functionalities represented by a prototype where a prototype is described as an object that represents the functionality but not the appearance of a finished artifact which can be used as a proof that a certain theory or concept or technology works or otherwise (Holmquist 2005).
To be able to verify a system's behavior in real time, there is need for an operational model of the system; an operational model in this context is one that can be executed in a suitable software environment (Bruno and Agarwal 1995). Analysis of traces generated from such executions can give further insights into the system's behavior as well as point out certain inconsistencies, missing requirements, verification of timing correctness in real-time systems etc. Using appropriate model-driven software engineering techniques, such executable programs to enact systems' properties can be synthesized from models created in some modeling environments. But before then, we must address questions such as "what should be the structure of the so-called operational program? What is the operational semantics of the chosen structure? ...". We try to address some of the possible questions with the framework proposed in this paper.
We propose an Object-Oriented framework that facilitates the synthesis (and specification) of operational (executable) representation of DES models for the enactment of such systems. In order to be general enough to accommodate a large category of DESs, our description of DES is guided by DEVS (Zeigler et al. 2000), a mathematical formalism that provide a sound basis for hierarchical description of DESs based on system-theoretic. Our choice of DEVS for generality is informed by the fact that it is considered to provide a common platform for describing most kinds of DESs and even approximated models for some kinds of non-discrete event systems (Vangeliswwe 2000). The challenge here, however, is that the operational semantics of DEVS is a simulation protocol while what we require is a semantic to drive the execution of the specified
behavioral processes. In this paper, we explore the mapping of DEVS concepts onto the Object-Oriented "Observer design pattern" (Gamma et al. 1994) to provide an execution semantics. We have chosen the observer design pattern to take benefit of its natural dialect for enacting the reactive systems and its ease of implementation in most general purpose programming languages. Of course it has some limitations that put its absolute suitability in question. We show in a later section, the measures we have taken to palliate some of these deficiencies (at least those that could have significant effects on the objective of the work).

We present overviews of DEVS formalism and relevant software engineering design patterns in the next section to set the scene for the reader to follow subsequent sections; then we compare and contrast the framework's intent and methodology with those of some related work and finally, afterwards, we present the essential elements of the framework followed by a case study to show its usability. We conclude the paper with discussions and perspectives.

BACKGROUND

Discrete Events System Specification (DEVS)

DEVS (Zeigler et al. 2000) is a system-theoretic mathematical formalism for specifying DESs as abstract mathematical objects for simulation. Basically, DEVS defines two abstraction levels for DESs - atomic and coupled DEVS. An atomic DEVS has a time base; state, input and output sets; and functions that define successive states and outputs events. A coupled DEVS on the other hand is an hierarchical composition of two or more atomic and/or coupled DEVS as components while specifying couplings between their input/output ports to enable their interactions.

Traditionally, DEVS exists in two major forms: classic DEVS (CDEVS) (Zeigler, 1976) and parallel DEVS (PDEVS) (Chow and Zeigler 1994, Chow, 1996), the main difference being that the latter supports concurrent state transition events within components of a coupled DEVS while the former does not. In this paper, we present PDEVS. Atomic DEVS (AM), which is defined as:

\[ AM = < X, Y, S, \delta_{\text{int}}, \delta_{\text{ext}}, \delta_{\text{conf}}, \lambda, ta > \] (1)

\[ X = \{(p, v), p \in I\text{Port} \land v \in \text{dom}(p)\} \] (2)

\[ Y = \{(q, v), q \in O\text{Port} \land v \in \text{dom}(q)\} \] (3)

\[ X \] and \[ Y \] are the sets of input and output events respectively. \[ I\text{Port} \] and \[ O\text{Port} \] are the sets of input ports and output ports respectively. \[ S \] is the set of states and at any given moment, a DEVS model is in a state \( s \in S \).

\[ ta : S \rightarrow R_{\geq 0} \land \lambda : S \rightarrow Y^b ; \delta_{\text{int}} : S \rightarrow S ; \]

\[ \delta_{\text{ext}} : Q \times X^b \rightarrow S \text{ and } \delta_{\text{conf}} : S \times X^k \rightarrow S . \]

\[ Q = \{(s, e) | s \in S, e \in [0, ta(s)]\} \]

The time advance function, \( ta \), maps each state to a duration after which an internal state transition, \( \delta_{\text{int}} \), is automatically fired. The external transition function, \( \delta_{\text{ext}} \), where \( Q \) is called the set of total states and \( e \) is the elapsed time since the last state transition, defines the system's response to an input event when the time advance of the current state has not expired. If the input event coincides with the expiration of

the time advanced, the confluent transition function, \( \delta_{\text{conf}} \), is invoked instead. The function \( \lambda \) defines the outputs that may accompany internal and/or confluent state transitions.

Coupled DEVS, CM which is defined as:

\[ CM = < X, Y, D, \{M_d \}_{d \in D}, EIC, EOC, IC > \]

\[ EIC = \left\{ \left( (M_d, ip_d, (d, ip_d)) \right) \middle| \right. \left( ip_M \in IPorts_M, ip_d \in IPorts_d \right) \]

\[ EOC = \left\{ \left( (d, op_d, (M, op_M)) \right) \middle| \right. \left( op_M \in OPorts_M, op_d \in OPorts_d \right) \]

\[ IC = \left\{ \left( (a, op_a, (b, ip_b)) \right) \middle| \right. \left( op_a \in OPorts_a, ip_b \in IPorts_b \right) \]

\( X \) and \( Y \) are as defined in (2) and (3) respectively. \( D \) is the set of component names with the specification of component \( d \in D \) represented by \( M_d \). \( EIC \) is the external input coupling relation, \( EOC \) is the external output coupling relation and \( IC \) is the internal coupling relation. The reader may consult (Zeigler et al. 2000) for further details about DEVS formalism and its operational semantics.

Object-Oriented Design Patterns

Design patterns in Object-Oriented modeling are documented solutions to some general problems that can be reused to build new models. In this subsection, we present the overviews of two design patterns from (Gamma et al. 1994) that are re-used in later sections to define the metamodel of our enactment framework.

Observer Design Pattern

It is a behavioral pattern for establishing relationships between objects at runtime such that changes in the state of an object (referred to as subject) trigger some actions in another (the observer). It is defined by the Gang of Four (Gamma et al. 1994) as a pattern that "define a one-to-many dependency between objects so that when one object changes state, all its dependents are notified and updated automatically."

Figure : shows an overview of the observer pattern. The basic idea is that the Subject maintains a list of references to some independent objects called the Observers. Whenever there is a change of state in the subject, all its observers must be notified by the invocation of the update method of each of them. Each observer (i.e., ConcreteObserver) must implement its update method to implement the corresponding actions to be taken whenever this happens. This pattern is widely used in Graphical User Interface (GUI) programming and it provides the underlying principle for the Model-View-Controller (MVC) architecture (Krasner and Pope 1988) so that all views are automatically updated whenever there is a change of state in the model.

Command Design Pattern

The command design pattern is shown in Figure 2. A command in this context means a method call. The pattern provides a methodology to encapsulate a command in an object and issue it (the command) in such a way that the requested operation and the requesting object do not have to know each other.
features of timed Petri nets, and workflows to model event-driven distributed systems. PROT nets describes a system as consisting of interacting autonomous objects called "actors" where each actor is an instance of a class. The behaviour of a class is described in a Petri nets dialect as consisting of places (describing the states) and transitions through which places are connected with arcs. An active place has a queue of message-carrying tokens that are moved from places to places through transitions. Some places are designated for Input/output operations to allow actors to interact with one another. Message passing between actors is achieved by moving tokens between their I/O interfaces. According to the authors, operational program codes can be generated from PROT nets specifications for general purpose languages though it is not clear what the structures of such codes look like. The similarity between PROTOB approach and the enactment framework presented in this paper is that the system description in both cases are based on some well defined formalisms - Petri nets in PROTOB and DEVS in our framework. Interestingly, the approach proposed in this paper can arguably accommodate a broader category of DESs based on the fact that the underlying formalism, DEVS, has been proven to provide a common denominator for most DES formalisms including Petri nets (Vangheluwe 2000).

Another interesting work that is related to that presented here is the one discussed in (Hu and Zeigler 2004). It proposed an approach of Model Continuity to Support Software Development for Distributed Robotic Systems based on Modeling-Simulation-Execution methodology (Hu and Zeigler 2002). As defined by the authors, Model continuity refers to the ability to use the same model of a system throughout its design phases, provides an effective way to manage this development complexity and maintain consistency of the software. Model continuity is ensured by using the same model in modeling, simulation and execution phases. Real-Time DEVS and Dynamic Structure concepts are used in modeling phase in order to support the modeling of the robots sensors and actuators as activities and dynamic reconfiguration of robots. In the simulation phase, different DEVS simulator implementations (supporting different communications schemes from point to point socket communication to advanced middleware such as CORBA) are used for the incremental verification of the model. The real-time execution is achieved by mapping the robot specifications into a real hardware execution environment controlled by DEVS real-time execution engine. It is, however, not clear what is the methodology used in building the said execution engine. The main similarity with this work and that presented in this paper is that the system description is based on DEVS-based systems in both cases. It is however different from ours in that the enactment engine proposed in this work resides in hardware for enactment of robot systems while ours is a software enactment on any suitable system.

**ENACTMENT FRAMEWORK FOR DES**

The methodology we propose is to express DEVS-based concepts using the dialect of the observer design pattern for the purpose of enactment. We do this by registering system ports as observers of other ports that may influence them. However, we acknowledge the fact that the notification process in the observer pattern poses some undesired effects

---

**RELATED WORK**

PROTOB (Baldassari et al. 1989; Baldassari and Bruno 1991) is a system development environment that integrates tools, for modeling, prototyping and implementation of distributed systems using an operational software life cycle paradigm. In PROTOB, systems are described with PROT nets, an Object-Oriented formalism that combines high-level
during the exchange of messages between ports; the processes of the system sending the message will be blocked until the receiving system finishes treating the message received. The effect is even more complicated when there is a cascade of notifications.

This is due to the synchronous calls to the update methods of the observers. We have tried to address this problem by using the command pattern to decouple the subject from its observer during notifications.

Figure 3 shows our attempt to introduce an asynchronous message passing between the subject and its observers to make it more suitable for enacting systems’ behaviors in real time. Compared to the command pattern presented in Figure 2, Subject, Observer, Notifier, Notification and ConcreteNotification are equivalent to Client, Receiver, Invoker, Command and ConcreteCommand respectively.

Therefore, the subject will delegate the notifications of observers to Notifier and continue its activities. Since the subject does not expect any return value from these method calls, it is easy to just use the "fire and forget" approach. Notifier has a pool of threads to which the requests are assigned on arrival, hence it does not have to always create threads thereby minimizing the overhead that may be incurred due to thread creation.

All methods in the AtomicSystem and CoupledSystem classes are abstract; therefore the concrete atomic and coupled system classes using the framework must implement them to provide the specific elements of the system being modeled. The update method of the AbstractSystem class has the implementation of the enactment protocol (to be provided in the next sub-section) which calls the user-defined functions when they are needed. The doInternalTransition, doExternalTransition and doConfluentTransition allow the user to describe the internal, external and confluent transition functions respectively. setCurrentStatus method is used to define the system’s states based on the instantaneous values of the state variables to be defined by the user in the concrete class. Similarly, mapTimeAdvance and mapOutputEvents methods must be implemented to provide the time advance and output functions respectively. Method mapActivities can be used to define the activities to be enacted for each state during execution. An activity is a set of operations that do not lead to change in state variables, reception of inputs and output events.

Coupling between any two ports in the in the CoupledSystem is realized by adding the target port to the list of observers of the source port.

**Enactment Protocol**

A transition in the state of an AtomicSystem can be triggered by a timed event (an automatic notification from the clock when the time advance of the current state has elapsed), an input event (a notification from an input port upon receipt of a new value) or both. By default, an AtomicSystem is a registered observer of its clock and all its input ports, so this allows for automatic notifications from both sides. In any case, an event is an object that encapsulates a message/value and information about the nature of its source, whether a port or clock. When the system receives notifications, all events received are stored in the event bag (eventBag) of the system. Then the system’s reaction will depend on the content of the bag.

If the event bag contains a time event, then it sends outputs (if any) to the appropriate output port(s) and then check if there are also input events in the bag. If a port event is found, then the doConfluentTransition method is invoked, otherwise doInternalTransition method is invoked. If the event bag contains only input events, then doExternalTransition method is invoked.

**Implementation**

We have implemented the framework’s metamodel and enactment protocol in Java. To use the framework, we can simply create classes inheriting from the AtomicSystem and CoupledSystem classes of the framework to get the skeletons the appropriate system unit. The user only needs to specify the properties that are peculiar to the system under study while the enactment mechanism is driven by the framework. Based on this implementation, we present a simple case study in the next section to illustrate its use.
CASE STUDY

We present a small example of a traffic light control system to illustrate the extension of the enactment framework for real-time execution of DES. The system consists of two components, control and display. The four states of the control, their durations and the corresponding light color to be on the display unit are summarized in Table 1. The control unit has only one output port which is connected to the only input port of the display unit as described in Figure 5. Whenever, there is a change in the state (internal transition) of the control unit, it sends an output which is received by the display unit to show the appropriate light color to the road user.

<table>
<thead>
<tr>
<th>Control states</th>
<th>Duration of control state (units)</th>
<th>Display color</th>
</tr>
</thead>
<tbody>
<tr>
<td>ready</td>
<td>3</td>
<td>yellow</td>
</tr>
<tr>
<td>moving</td>
<td>10</td>
<td>Green</td>
</tr>
<tr>
<td>braking</td>
<td>3</td>
<td>Yellow</td>
</tr>
<tr>
<td>stopping</td>
<td>5</td>
<td>Red</td>
</tr>
</tbody>
</table>

The specification of the system is presented in Figures 6-9. The control unit is shown in Figure 6. It is an atomic unit, so it has to extend the framework’s AtomicSystem class which provides the required system-specific methods to be completed as indicated by the methods with @override annotation in Figure 6. The single output port is created using the addOutputPort method provided by the framework.

The display unit is presented in Figure 7. It is also an atomic unit and maintains only one state with approximately infinite time advance as indicated by the Long.MAX_VALUE in the computeLifeSpanFunction. So, it will never receive a time event since the time advance will never expire. Therefore,
only external transition is possible. Whenever, it receives an input event (which is an instruction from the control unit), it changes the color of light displayed to the new color received.

Figure 7: Display Unit of the Traffic Light System

Figure 8 shows the coupled system that has the control and display units as components. Being a coupled system, it extends the CompositeSystem class of the framework which provides the required methods to be completed. It basically creates and registers its components and establish any coupling(s) between them. In this case, there is only one coupling between the components as shown. Figure 9 shows an excerpt from the result of the enactment of the specification. The first column of the result shows the wall clock time, the second column shows the identity of the component in context and the third column shows the event being reported. Note that each component has its clock that monitors its activities based on the ticks of the wall clock.
CONCLUSIONS

We have presented an Object-Oriented framework for the enactment of DESSs. The framework provides a template to guide the synthesis/writing of program codes from DEVS-based models and the protocol for real time enactment of system's behavior. The main idea is to be able to generate or specify an operational model in form of software systems to verify and validate the real time behavior of system models using wall clock time as reference, i.e., the scheduling and execution of events are done based on the physical time. We used the Object-Oriented Observer design pattern to express DEVS-based system constructs by mapping the system's structural and behavioral properties to the structure and semantics respectively of the observer pattern. The subject-observer relations are used to establish couplings between ports of the components of a system while the notification mechanisms are used to trigger state transitions. We provided a Java implementation of the framework and a
case study to illustrate its use to specify and enact discrete events systems.
In future research, we intend to provide a standard format for the traces of the execution so that it can be amenable to further rigorous analysis. We also intend to integrate the framework with compatible model-based development environments to extend the kinds of analysis they provide. With appropriate applications of model-driven technologies, we can synthesize enactment codes based on the template provided from any suitable DEVS-based model.

REFERENCES


Hu, X. and Zeigler, B. P. 2002. *An integrated modeling and simulation methodology for intelligent systems design and testing*. ARIZONA UNIV TUCSON.


HARDWARE OPTIMIZATION
SIGNAL PROCESSING AT DIFFERENT MENTAL TASKS EXECUTION FOR BRAIN-COMPUTER INTERFACE

Svetla Radeva
Department of Information Technologies
University of Telecommunications and Post
1st Acad. Stefan Mladenov Str., 1700 Sofia, Bulgaria
e-mail: svetla_ktp@abv.bg

Dimitar Radev
Department of Telecommunications
University of Telecommunications and Post
1st Acad. Stefan Mladenov Str., 1700 Sofia, Bulgaria
e-mail: dradev@abv.bg

KEYWORDS
Brain-Computer Interface, Neural Networks, Human–Computer Interaction, Communication and Control

ABSTRACT
The brain-computer interface (BCI) aim to use Electroencephalography (EEG) activity or other electrophysiological measures of brain functions as new non-muscular channels for control and communication with smart devices and smart mobile applications for disabled persons. The research aims developing of technology for communication with smart mobile applications, based on processing of recorded electrophysiological signals at execution of different mental tasks. The interface use depends on the interaction of two adaptive controllers: the user, who generates brain signals that encode intent sitting on his invalid chair and the computer system that translate these signals into commands that accomplish the user’s intent for connection with smart mobile applications. Brain Computer Interface, controlling computer using brain directly, is a Human-Computer Interaction (HCI) system which is in developing state and has very promising features for persons who cannot access traditional computer system due to their physical disabilities. In this research the recorded with experimental setup electrophysiological signals for execution of three different mental tasks after noise filtering are estimated on the base of clustering and classification with Bayesian Network classifier and pairwise classifier.

INTRODUCTION
The brain-computer interface (BCI) aim to use electrophysiological measures of brain functions as new non-muscular channels for control and communication. Developing such new communication and control technology will improve the life of disabled persons with neuromuscular disorders, as brain stem stroke, amyotrophic lateral sclerosis and spinal cord injury. The brain-computer interfaces gives to these users basic communication capabilities and they can express their desires to caregivers, or to operate word-processing programs or neuro-prostheses (Allison et al. 2007).

The electrical fields produced by brain activity can be recorded from the scalp (EEG), from the cortical surface (electrocorticographic activity, (EcoG)), or from within the brain (local field potentials (LFPs)) or neuronal action potentials (spikes)) (Shih et al. 2012).

One of advantages of EEG recording is that it is easy and non-invasive, but a disadvantage of EEG is limited topographical resolution and frequency range (Millán et al. 2007). The electrocorticographic activity EcoG has better topographical resolution and frequency range, but his disadvantage is that it requires implantation of electrode arrays on the cortical surface, which has been done only for short periods in humans (e.g., a few days or weeks) (Wolpaw, J. and E. Wolpaw 2012).

Recording within other brain structures (Intracortical recording) provides the highest resolution signals, but it requires insertion of multiple electrode arrays within brain tissue and faces (Fairclough 2009).

In this research we focus our activity on modeling and simulation of three mental tasks for developing of brain-computer interface. This interface will help disabled persons to connect with smart mobile applications.

BRAIN-COMPUTER INTERFACE
Brain-computer interface applications has wide scope of implementation as human computer interaction technique. First of all, BCI makes computer system fastest because brain-computer interface does not contains manual information transfer at all. That fact makes the fastest interface very promising to all domains but there are some specific domains where this interface is specifically very important, like computer games, which can be made more attractive, useful and effective with BCI; embedded systems; using BCI in operating machines; medical industry as a biggest area of BCI application etc. (Loke et al. 2007).

BCI system for smart mobile applications consists of the following components, realized on the following steps:

• **Step 1:** Thinking in Brain - when something is to be done a thought is developed into the brain which leads to development of a neuron potential pattern.

• **Step 2:** Reading Brain by EEG - when the
developed potential pattern is read by EEG (or other similar techniques) to be transformed into an analyzable signal patterns. This is also known as EEG spectrum.

- **Step 3:** Analysis of EEG spectrum - the signal pattern developed by EEG equipment is analyzed using various pattern analysis techniques.

- **Step 4:** Recognizing EEG spectrum - based on the signal analysis we recognize what task brain wants to get from computer or mobile device.

- **Step 5:** Converting into suitable computer signal - once we know the task to be done we can easily determine proper computer command (or sequence of command i.e. program) to get the task from computer or mobile device.

- **Step 6:** Sending the signals to computer system - after discovering the required command or program, send the same to CPU which then execute the required task.

- **Step 7:** Feedback to the user - after CPU accepts the input it carries out the operation and sends the feedback to user in various feedback-forms e.g. video, audio etc.

BCI implementation is connected with some limitations and difficulties connected with appearing of noises during EEG, signal acquisition, clustering of neurons and user interface design.

**SIGNAL PROCESSING AND NOISE FILTERING OF RECORDRD BRAIN SIGNALS**

A BCI records brain signals and processes them to produce device commands. This signal processing has two stages. The first stage is feature extraction, the calculation of the values of specific features of the signals. These features may be relatively simple measures such as amplitudes or latencies of specific potentials (e.g., P300), amplitudes, or frequencies of specific rhythms (e.g., sensorimotor rhythms), or firing rates of individual cortical neurons, or they may be more complex measures such as spectral coherences. To support effective BCI performance, the feature-extraction stage of signal processing must focus on features that encode the user’s intent, and it must extract those features as accurately as possible.

At any moment the human brain generates wave for a particular thought, but at the same time generates also some waves corresponding to other unnecessary thoughts. These additional waves act as noise for original waves. For handling these noise-waves the user have to increase his concentration during the BCI process. But sometimes it is not possible to concentrate fully on a particular thought. For handling this problem it is necessary to develop some noise filtering mechanism that can detect the unrelated spectrum and filter them out from the useful spectrum.

The second stage is a translation algorithm that translates these features into device commands. Features such as rhythm amplitudes or neuronal firing rates are translated into commands that specify outputs such as cursor movements, icon selection, or prosthetic operation. Translation algorithms may be simple (e.g., linear equations), or more complex (e.g., neural networks, support vector machines) (Müller et al. 2008).

To be effective, a translation algorithm must ensure that the user’s range of control of the chosen features allows selection of the full range of device commands (Fogarty et al. 2005). For example, suppose that the feature is the amplitude of a 8–12 Hz mu rhythm in the EEG over sensorimotor cortex; that the user can vary this feature over a range of 2–10 V; and that the application is vertical cursor movement. In this case, the translation algorithm must ensure that the 2–10 V range allows the user to move the cursor both up and down. The translation algorithm should have the capacity to encourage improvements in the user’s control. For example, if the user’s range of control improves from 2–10 to 1–15 V, the translation algorithm should take advantage of this improvement to increase the speed and/or precision of cursor movement control.

At any moment the human brain generates wave for a particular thought, but at the same time generates also some waves corresponding to other unnecessary thoughts. These additional waves act as noise for original waves. For handling these noise-waves the user have to increase his concentration during the BCI process. But sometimes it is not possible to concentrate fully on a particular thought. For handling this problem it is necessary to develop some noise filtering mechanism that can detect the unrelated spectrum and filter them out from the useful spectrum.

The process of noise filters design requires some situation and application of specific knowledge from neurology. First we study the situation and then identify which part of brain will deal with it. Afterward we carry out some close to ideal situation where user tries to concentrate fully on the application. We carry out some such experiment for finding out the pattern of the signals and on the basis of this we prepare and calibrate our noise filter. This problem is very difficult and requires a lot of time for solving it.

**CLUSTERING OF NEURONS**

The signal processing of BCI records is connected with clustering of neurons, where we have to divide 80-120 billion brain-neurons into few clusters and the big question is – on what basis we should divide the neurons? There is no exact answer on this question and the problem can be solved only experimentally.

There exists no standard way to define the exact clustering which will be perfect for all situation. There exists some systems which are very suitable for large number of problem but still a lot of research is required in this field for automatic clustering of neurons. For solving this problem is involved Artificial Intelligence and Artificial neural network. Although, there exists several alternatives for pattern analysis and recognition, Artificial Intelligence and Artificial Neural Network
provides very effective and useful algorithms for this (Poolman et al. 2008).

**EXPERIMENTAL MEASUREMENTS AND SIMULATION**

The experimental brain-computer interface setup with smart device communication was worked out at the Telecommunications and Post University at Sofia. The experimental BCI system includes 3D camera Panasonic HDC-Z0000, sender Spectrum DX9 DSMX, Sony GoPRO – GoPro HERO3, Nikon D902D smart TV Samsung UE-65HU8500 + LG60LA620S, ACER K11 Led projector, Linksys EA6900 AC1900 smart router, Pololu Zumo Shield, 8 core/32GB RAM/4TB HDD/3GB VGA computer for video processing that translate EEG signals into computer commands. Two Electro-Caps (elastic electrode caps) was used to record each from positions C3, C4, P3, P4, O1, and O2, defined by the 10-20 System of electrode placement at experimental setup. The 10-20 System is an international standard for EEG electrode placement locations on the human scalp. The system defines a grid relative to physical landmarks on the head, such as the indentation between the nose and forehead (nasion), and the bump on the back of the head (inion) at the occipital protuberance as shown in Figure 1. Electrode locations are defined by either 10% or 20% increments between these landmarks.

**Figure 1. Electrode placement on electro-caps**

The electrodes were connected through a bank of Grass amplifiers and band pass filtered from. Data was recorded at a sampling rate of 250 Hz with a Lab Master 12 bit A/D converter mounted in computer. Eye blinks were detected by means of a separate channel of data recorded from two electrodes placed above and below the subject's left eye. Data was recorded for 14 seconds during each task and each task was repeated 10 times per session. The subjects were asked to perform the following three mental tasks:

- baseline task - for any possible subjects relaxed and not thinking activity;
- emergency call - subjects imagine to dial up 122;
- music – subject switch on radio from mobile device.

For signals registration was used time-domain regression method, which helps to remove the blink artifact noise contained in the original EEG signals. Aiming at different leads, eye signals multiplied by corresponding weighting factors to modify amplitude, and then the EEG signal subtract the modified eye signals to reach the purpose of removing the strong eye electricity. The EEG signals are processed as typically non-stationary random signals.

For simulation the recorded EEG data are segmented by rectangular windows. The length of each window is 1s (200 sampling points). Each mental task is repeated 10 times. Each time lasts 14 seconds. Each channel records 3000 sample data for each test.

Based on results from pilot recordings, we selected the parietal (P3 and P4) regions as the locations of interest, with both electrode references tied on top of the head at the central region (Cz), shown on Figure 1. Tying the references for the two EEG channels together allowed us to make meaningful comparisons between the values from each channel. The experimental setup also includes a ground electrode connection that is attached to an ear lobe. The purpose of grounding is to provide electrical protection that prevents damaging the sensitive inputs of the device. Its specific location on the head does not impact the recorded signals.

**DATA ANALYSIS AND SIMULATION RESULTS**

The signals measured from considered brain-computer interface was classified with Bayesian Network. For this classification was performed some basic signal processing to transform the time series data into a time independent data set. On the base of this was computed a set of base features that was mathematically combined to generate a much larger set of features. This feature selection was processed to prune the feature set, keeping only those that added the most useful information to the classifier and to prevent overfitting. Selected features were used to train a Bayesian Network and perform the classification.

The first 4 seconds from each task recording was removed to avoid unwanted response artifacts into the EEG data. This is sufficient time for both the computer and the experimenter prompts to complete and allows a moment for task onset to occur within the participant. The remaining 10 seconds then contains only signals during which the task was actually being performed.

The data were converted into a time independent dataset as the most machine learning algorithms do not handle time-series data. For this purpose, the EEG signals were sliced into small overlapping windows and then was computed features based the content of each window. Specifically, we divide each 14-second task recording into 2-second windows overlapping by 1 second. This provides 13 windows per task period. This set of
windows becomes the set of training instances used by the machine learning algorithm for constructing and validating the classifier model (Müller et al. 2008).

The process of EEG data analysis is connected with the spectral power of the signal in a set of six standard frequency bands which have been observed to correspond with certain types of neural activity. These frequency bands are: 1-4Hz (delta), 4-8Hz (theta), 8-12Hz (alpha), 12-20Hz (beta-low), 20-30Hz (beta-high), and 30-50Hz (gamma).

After a Fourier transform of the EEG data was received the frequency content of the signal. The data recorded from BCI EEG has very little signal above 50Hz with the exception of 60Hz interference from indoor power lines. As a result, the frequency range used for gamma band can be reduces.

For each of selected windows was computed the following: signal power in each of the six frequency bands for each channel, phase coherence (similarity in mean phase angle) in each band across channels, and each band power difference between the two channels. This results in 24 features that are commonly used for EEG signal analysis. In addition to these features, we also compute the following set of more general signal properties for each input channel:

- mean spectral power;
- peak frequency;
- peak frequency magnitude;
- mean phase angle;
- mean sample value;
- number of samples above zero;
- zero-crossing rate;
- mean spectral power difference between two input channels;

In total, this provides with 312 base features for each window. Non-linear manipulations of features such as this is a common machine learning technique used to compensate for a potential lack of expressiveness in the statistical model used by the classifier.

The input variables were divided into discrete bins that might provide meaningful discriminations in the output variable. This helps to find correlation between basic parameters of different mental tasks and for each task evaluate sub-sets of features favoring those that have a high correlation with the output variable while having low inter-correlation among the features within the selected set. This provides computationally inexpensive method of identifying a relatively small subset of useful features for the classification problem.

On average, this process reduced the number of features to 126 for the 3-task classifier and 63 features for the pair-wise classifier. We then applied a more computationally expensive wrapper-based feature selection process. In this case a classifier model begins with an empty set of features and then incrementally adds or removes features based on their impact on overall classification accuracy. This further reduced the number of features used for classification, resulting in an average of 54 features for the 3-task classifier and 27 features for the pair-wise classifier.

After performing the feature selection procedure for each participant, was used a Bayesian Network classifier to identify which task was being performed during a given test window. In this work was used 18-fold cross validation, instead of standard 10-fold cross validation, to control the block design of the data collection procedure.

For each fold, the model was trained on 9 of the 10 available trials and reserved one trial for testing. A trial contains 13 contiguous windows for each task. Each of reported results is the mean classification accuracy after repeating this process 10 times using a different test trial for each fold. This is more representative of the performance that we would expect if a new trial was recorded and tested with proposed 18-fold cross validation than if was used standard 10-fold cross validation. Table 1 presents received results of the classification accuracies for each of 6 subjects.

<table>
<thead>
<tr>
<th>Subject number</th>
<th>Mental Tasks</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Base</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>91.3%</td>
<td>62.6%</td>
</tr>
<tr>
<td>2</td>
<td>92.4%</td>
<td>81.7%</td>
</tr>
<tr>
<td>3</td>
<td>87.8%</td>
<td>71.3%</td>
</tr>
<tr>
<td>4</td>
<td>90.2%</td>
<td>83.6%</td>
</tr>
<tr>
<td>5</td>
<td>93.7%</td>
<td>87.6%</td>
</tr>
<tr>
<td>6</td>
<td>89.3%</td>
<td>78.4%</td>
</tr>
<tr>
<td><strong>Mean</strong></td>
<td><strong>90.8%</strong></td>
<td><strong>80.9%</strong></td>
</tr>
</tbody>
</table>

The Bayesian Network classifiers for five mental tasks yield classification accuracies depending on the user into the interval between 71.2% and 87.6%.

The prior for these classifications, or the expected result of a random classifier, is 33.3%. The pair-wise classifiers have a prior of 50% and yield accuracies of between 85.4% and 94.7%.

Comparison between received results with Bayesian Network classifiers and pair-wise classifier show that the variance between parameters for Bayesian Network classifier $q(x)$ will decay slower than parameters of pair-wise classifiers $p(x)$.

Here $p$ and $q$ are distributed with normalization and the variance of $q$ for $x \to \infty$, $q(x)$ will decay slower than $p(x)$, as is shown in Figure 2, where $p(x)$ is lower line and $q(x)$ is middle line and $p(x)q(x)$ (upper line) is shown to justify numerical evaluation for all confidential interval and to approximate the expectation. In Figure 3 by shifting the mean of the sampling distribution is seen the
results of choosing a Bayesian Network classifiers $q(x)$ (right line) with a center of mass far away from pairwise classifier $p(x)$ (left line).

Figure 2. A plot of $p(x)$, $q(x)$ and $p(x)f(x)$.

Once the classifier had been trained, classification of new test data occurred nearly instantly allowing it to be used in a real-time implementation. The classifier model could then be updated periodically in the background given the availability of new training data. The effective data rate of such a system would be approximately 10-30 bits/min depending on the degree of averaging, which corresponds to the level of noise filtering in the data stream.

CONCLUSIONS

In this research is suggested an approach for human – computer interaction with classification of recorded electrophysiological signals at different mental tasks for connection via BCI with smart mobile applications. With considered experimental setup of brain-computer interface were provided experiments with six subjects for execution of three mental tasks. The measured outputs were classified with Bayesian Network classifier and with of pair-wise classifier the received results are compared and shown.

The future implementation of BCI with smart mobile applications is connected with three main concepts:

1) increasing the bandwidth between human and computer and the effectiveness of that bandwidth;

2) enhancing or predicting the comprehension of information in context;

3) supporting the formation of ideas.

BCI technologies as a very promising approach can also provide the potential for users to actively modify their own brain processes or states. Methods such as neurofeedback can already allow for individuals to adjust their own brain function in an attempt to attain a more desirable state.

ACKNOWLEDGEMENT

This paper is a part of a research project of the Scientific Research Fund at the Bulgarian Ministry of Education and Science DFNI 102/2014 “Human-computer interface for medical assisting systems for life improving of people with propelling problems”. The authors are thankful to the Scientific Research Fund at the Bulgarian Ministry of Education and Science for financial support.

REFERENCES


A Based-Optimization Scheduling Approach for Multi-processor Platform with DVFS Facilities

Hamza Chniter  
LISI Lab, INSAT, Carthago  
University, Tunisia  
FST faculty - University of El-Manar, Tunisia  
hamza.chniter@yahoo.fr

Fethi Jarray  
ICEDRIC Laboratory, CNAM,  
292 street St-Martin 75141, Paris  
cedex 03, France  
fethi_jarray@yahoo.fr

Mohamed Khalgui  
SystemsControl Lab, Xidian  
University, China,  
LISI Lab, INSAT, Carthago  
University, Tunisia  
khalgui.mohamed@gmail.com

Maryline Chetto  
IRCCyN Laboratory  
University of Nantes, France  
Maryline.Chetto@univ-nantes.fr

KEYWORDS
Embedded systems, Reconfiguration, Real-time and low-power scheduling, Multi-processor, Optimization, Integer programming, Heuristic

ABSTRACT
In this paper, we address the problem of how to allocate a set of periodic tasks when new ones are added at runtime due to reconﬁgurations on multi-processor systems with DVFS (abrev, Dynamic Voltage and Frequency Scaling) capabilities. When new tasks are added, they may not be feasible to be scheduled with the original processing speeds of processors. We develop a mechanism to increase these speeds so that the tasks can still meet their deadlines, and propose an approach to find the allocation and WCET of the tasks in order to optimize the energy consumption and makespan while meeting temporal constraints. More speciﬁcally, we propose a combinatorial optimization method based on integer programming and heuristics, and introduce an additional approach based on deadline adjustment used when no available speed can fulﬁll the objectives. We present the results of a simulation study that shows the benefits of our assignment method in terms of some metrics such as time, energy and makespan.

INTRODUCTION
An embedded system (ES) is a device which is dedicated to specific functions. It includes hardware and software parts which are designed to operate without human intervention (1). Modern embedded systems are often based on micro-controllers and integrated into more complex systems to perform time-sensitive and specific applications. They are usually linked with real-time constraints that determine their reliability and accuracy (2). A real-time embedded system is any system whose role is to control its environment while respecting different functional and extra-functional constraints (3). The real-time property is assigned if the accuracy of the system is determined by the dates on which the execution results are available. Furthermore, the notion of valid result becomes true if it is correct and available in the time interval set by the environment. These ES must be able to react and solve temporarily any disturbance due to a run-time reconfiguration of the system. A reconfiguration is any operation allowing the addition-removal-update of OS tasks in order to adapt the system to its environment. Some important applications impose temporal constraints on the response time while running on systems with limited power resources (4). In this context, Several studies and solutions based on DVFS technology (3) have been performed. These solutions aim to minimize the system energy consumption by adjusting the working voltage and frequency of the processor. For this reason, two combinatorial optimization solutions based on integer programming and heuristics have been developed and presented in (5) and (6). The first exposes an integer programming model and a heuristic which try to act to the processor speed during the execution of the OS tasks. The second also includes a mechanism based on deadlines adjustment to support the issue when all the available speeds of the processor are unable to satisfy the requirements and the system lies infeasible. Experimental results have shown that our approaches perform well in terms of maintaining the stability of the system and minimizing the energy consumption. Recently, these approaches have been implemented in a real-time middleware called RT-MED which corresponds to a software layer to be placed above any operating system (7). The basic contribution in the current paper is a continuation of these works. It aims to generalize the presented solutions for multi-processor platforms. Thus, the target is how to execute a set of OS tasks on set of processors by taking into account a set of optimization criteria. OS tasks can be spread across several processors and thus be executed more quickly than on a single processor. The common goal is to minimize the run-time of a set of tasks. Besides, many criteria can be added such as minimizing the cost, the communication delay and the energy consumption. In this contribution, the target architecture is a multiprocessor with the support for dynamic voltage scaling. We reformulate the problem and propose two oriented-optimization approaches to solve it: Integer programming and simulated annealing. These approaches support an appropriate mechanism to manage the tasks
scheduling throughout execution. At first, they are responsible for the allocation of all tasks to the different available processors to be executed. Then, these approaches must defend any failures in the system caused by the reconfiguration scenario which can undergo during the execution. They touch on the one hand, the hardware part of the system by changing the execution frequencies of a set of tasks in order to find an optimal combination between the different processors, tasks and frequencies. This combination can meet all timing constraints and ensure the optimal energy consumption and makespan. The makespan is defined as the completion time of the last task in the schedule sequence (8). On the other hand, the proposed methods provide proactive solutions when the maximum frequencies of the processors can not overcome the problems of feasibility. The solutions of the proposed methods were compared to those obtained in other works such as (28, 25, 29). The result shows that IP and SA give optimal and efficient results compared to the related works according to different metrics.

The remainder of this paper is organized as follows. We discuss in Section 2 the originality of this work by studying the state of the art. In Section 3, we expose the problem. We present in Section 4 the contribution dealing with an integer program formulation and a simulated annealing to solve the problem. Finally, numerical results are depicted and discussed in Section 5.

RELATED WORKS

The main challenge of scheduling problems upon multi-processor platforms is to assign tasks to the processors and determine their execution sequences on each processor (9). Therefore, multi-processor scheduling problems generally seem to be harder than single ones (10). There are various papers in this context. The common objectives are the minimization of completion time, tardiness, and makespan. (11) studies the problem of scheduling of independent jobs with sequence-dependent setup times and job splitting, on a set of identical parallel machines such that the maximum completion time is minimized. (12) proposes a branch and bound algorithm for the identical parallel machine scheduling problem with the objective of minimizing total tardiness. (13) deals with the scheduling problem of identical parallel machines. It formulates a Mixed Integer Programming model for the problem and proposes two meta-heuristics: Simulated Annealing and tabu search. Several DVFS-based solutions are also proposed for real-time embedded systems running on multi-processor platforms (14). These studies aim to solve the allocation of tasks to the available processors with the run-time voltage scaling for each processor. There are also various studies on energy-efficient scheduling for embedded systems in multi-processor platforms (23). Here, the scheduling process aims to allocate tasks to a set of processing elements by using the lowest voltage under different constraints. Scheduling tasks on multi-processor platforms is an NP-complete problem (15). Many contributions use heuristic approaches to achieve optimal solutions. (24) investigate a heuristic task allocation strategy to solve the energy latency for tasks in heterogeneous embedded systems. (16) and (22) present three resource allocation policies in heterogeneous systems. Yet, these scheduling strategies give high performance for non-real-time applications because they cannot respect the temporal requirements of real-time tasks. (17) presents a hard real-time scheduling with energy constraints and provides a collection of tools for automatic assistance in design activities. (18) develops both static and dynamic scheduling schemes based on DVFS technique to maximize overall reliability within energy and timing constraints. (19) presents a scheduling algorithm for energy harvesting real-time embedded systems with a realistic model for the battery charging and discharging processors.

The originality of our approach emanates from that they can provide not only snapshot and effective solutions but also optimal ones compared to the related works in (28, 25, 13, 29). Our solutions permit to manage different reconfigurations which are unpredictable and depend on the reaction between the system and its environment. The calculation proves that these approaches give better results than the work presented in (28) according to the computational time and knowing that this work does not address the context of reconfigurable systems which undergo specific reactions and events during executions. These reconfigurations often require snapshots interventions and treatments and therefore extra time calculation. The results presented in (28) does not also include the constraint of energy which remains an advantage for our approaches. The comparison confirms also the performance of our work to provide the best optimal solutions compared to works in (13) and (25) either in the execution time or the makespan.

DEFINITION AND NOTATION

In this section, we first present the notation used for task and energy models. Then, we describe the feasibility test. Finally, we expose the basic problem which is treated in this paper with a case study to illustrate it.

Tasks and Energy Models

We consider a set of $n$ real-time tasks $T = \{T_1, T_2, \ldots, T_n\}$ to be executed upon a multi-processor platform. Tasks are periodic, independent and non-preemptive (29). Deadlines are equal to periods and multiprocessing is authorized. We assume the hardware platform to be composed of $m$ identical processors. Each one has $nbf$ available scaling factors. Each task $T_i$ can be assigned to at most one processor $p$. $T_i$ is character-
ized by the following parameters (2):

- Release time \( r_i \), i.e. \( T_i \) cannot start before time \( r_i \) on each processor,
- Absolute deadline \( d_i \), i.e. \( T_i \) must finish before time \( d_i \),
- Computation time at the normalized processor frequency \( Cn_{ip} \).

For each task \( T_i \), we define its utilization factor \( U_{ip} \), \( i = 1 \ldots n, p = 1 \ldots m \), as the ratio of execution requirement to its period in the processor \( p \): \( U_{ip} = \frac{C_{ip}}{d_i} \). We define the total utilization \( U_{tot} \) to be the sum of the utilization of all tasks in each processor: \( U_{tot} = \sum_{p=1}^{m} \sum_{i=1}^{n} \sum_{k=1}^{nbf} U_{ip}X_{ipk} \), \( X_{ipk} \) is a binary variable which is defined in equation (2). We also define the maximum utilization of the tasks \( U_{max} \) to be the largest utilization of any task in any processor. We respectively denote by \( f_n \) and \( V_n \) the normalized frequency and voltage of any processor \( p \). We assume that they are proportional. We suppose that \( T_i \) is executed at frequency \( F_{ip} \) and voltage \( V_{ip} \) in the processor \( p \). We denote by \( \eta_{ip} \) the \( k^{th} \) available scaling factor of voltage on the processor \( p \), \( V_{ip} = \frac{V_{ip}}{\eta_{ip}} = 1 \ldots n, p = 1 \ldots m \). So we get \( C_{ip} = Cn_{ip}\eta_{ip} \). When the system is running at frequency \( F \) and voltage \( V \), the power consumption is given by \( P = CV^2F \) where \( C \) is a constant that depends on the hardware circuit (2). The power \( P_{ip} \) consumed by task \( T_i \) on processor \( p \) is:

\[
P_{ip} = CV_{ip}^2F_{ip} = CV_n^2f_n\eta_{ip}^2/k^2.
\]

Consequently, the energy \( E_{ip} \) consumed by a task \( T_i \) on processor \( p \) is:

\[
E_{ip} = P_{ip}C_{ip} = CV_{ip}^2f_n\eta_{ip}^2 = K\frac{C_{ip}}{\eta_{ip}^2} \quad \text{where the constant} \quad K = CV_n F_n.
\]

So the total energy consumption in the system is:

\[
E = \sum_{p=1}^{m} \sum_{i=1}^{n} \sum_{k=1}^{nbf} E_{ip}X_{ipk} = K\sum_{p=1}^{m} \sum_{i=1}^{n} \sum_{k=1}^{nbf} \frac{C_{ip}X_{ipk}}{\eta_{ip}^2}
\]

(1)

Where

\[
X_{ipk} = \begin{cases} 
1 & \text{if task } i \text{ is assigned to processor } p \\
0 & \text{otherwise} 
\end{cases}
\]

The problem to be solved is as follows: how to dynamically assign the tasks to the processors in optimal calculation times and how to select the speed of each processor while minimizing the global energy consumption and meeting deadlines.

### Schedulability Analysis

Several schedulability analysis of periodic real-time tasks base on the EDF (Earliest Deadline First) policy have been presented. According to (20), the utilization guarantee for EDF or any other static-priority multiprocessor scheduling algorithm cannot be greater than \( \frac{k+1}{m+1} \) for a \( m \)-processors platform. (21) proved that it is possible to schedule on \( m \) processors, any system of \( n \) independent periodic tasks with maximum individual utilization \( U_{max} \) and total utilization \( U_{tot} < \frac{m+1}{k+1} \) where \( k = \frac{1}{U_{max}} \). When \( U_{max} = 1 \), the guaranteed utilization bound is \( \frac{m+1}{2} \).

### Issue

We assume a given reconfigurable real-time system composed of a set of \( n \) tasks and \( m \) identical processors. The system is feasible since all the tasks are allocated to the processors and meet the corresponding deadlines described by the user requirements. We assume adding a set of new tasks to the current feasible system. Finally, we suppose that the extended system of tasks is not feasible where some deadlines are violated and the energy consumption increases. The arising question is how to provide an optimal solution that will optimize the energy consumption and guarantee the respect of deadlines.

### Case Study:

We consider a real-time embedded system composed firstly of 5 tasks as depicted in Table 1. We consider a multi-processor platform composed of 3 processors. According to the feasibility test, \( \frac{1}{U_{max}} = \frac{99}{39} = 2.307 \). The system is feasible since the total utilization \( U_{tot} = 1.568 \leq \frac{(3+2.307+1)}{2.307+1} = 2.395 \). Energy is equal to 3384. Table 1) shows the allocation of the tasks to the processors (Grey columns describe the allocation plan). We assume a run-time reconfiguration scenario to add 3 new tasks in order to adapt the system to its environment under particular constraints (Table 2). \( \frac{1}{U_{max}'} = \frac{100}{85} = 1.235 \). The total utilization \( U_{tot}' = 3.610 \geq \frac{(3+1.235+1)}{1.235+1} = 2.105 \). The new system becomes infeasible since the timing constraints can not be met and the energy consumption increases to 8736. To overcome the problem, we propose a method to dynamically adapt, in optimal calculation time, the processor’s speed by using integer programming and simulated an-

<table>
<thead>
<tr>
<th>Task</th>
<th>CPU</th>
<th>Release WCET</th>
<th>Start time</th>
<th>Finish time</th>
<th>deadline</th>
</tr>
</thead>
<tbody>
<tr>
<td>T1</td>
<td>P1</td>
<td>0.00</td>
<td>0.00</td>
<td>2.00</td>
<td>70.00</td>
</tr>
<tr>
<td>T2</td>
<td>P1</td>
<td>0.00</td>
<td>2.00</td>
<td>4.00</td>
<td>80.00</td>
</tr>
<tr>
<td>T3</td>
<td>P2</td>
<td>0.00</td>
<td>0.00</td>
<td>3.00</td>
<td>90.00</td>
</tr>
<tr>
<td>T4</td>
<td>P3</td>
<td>0.00</td>
<td>2.00</td>
<td>4.00</td>
<td>110.00</td>
</tr>
<tr>
<td>T5</td>
<td>P3</td>
<td>0.00</td>
<td>0.00</td>
<td>3.00</td>
<td>100.00</td>
</tr>
</tbody>
</table>
nealing to meet deadlines and optimize the global energy consumption and the makespan. Our solution aims also to tune the deadlines of tasks to be compatible with available frequencies.

CONTRIBUTION: FEASIBLE RECONFIGURATION FOR MULTI-PROCESSOR EMBEDDED SYSTEMS

In this section, we introduce two distinct approaches for the online modification of processors speeds and tasks deadlines: Integer programming and simulated annealing. Our goal is to optimize the calculation time of solutions.

**Integer Programming Approach**

We suppose that each processor has a set of $k$ available scaling factors. We use a binary matrix $X = (X_{ipk})$ to model the system configuration.

$$X_{ipk} = \begin{cases} 
1 & \text{if the task } T_i \text{ is allocated to the processor } p \\
0 & \text{otherwise}
\end{cases}$$

The following constraints should be satisfied:

1. Each task is executed on one processor without preemption. To ensure only one active task at any time on each processor, we add $t_{ip} - t_{jp} \geq C_{ip} \eta_k X_{ipk} - M \alpha_{ij}$ and $t_{jp} - t_{ip} \geq C_{ip} \eta_k X_{ipk} - M(1 - \alpha_{ij})$ for every pair of tasks $T_i$ and $T_j$, where $\alpha_{ij}$ is a binary variable and $M$ is a big constant. $\alpha_{ij} = 1$ means that $T_j$ is executed before $T_i$. Such formula means that, if task $T_i$ starts before task $T_j$ on processor $p$, task $T_j$ cannot start execution on processor $p$ before task $T_i$ finishes execution.

2. The deadline of $T_i$ should be respected, $t_{ip} + C_{ip} \eta_k X_{ipk} \leq d_i$.

3. The release time of $T_i$ should be respected, $t_{ip} \geq r_i$.

4. $T_i$ must be executed with one and only one frequency, $\sum_{k=1}^{m} X_{ipk} = 1$

5. $T_i$ is allocated to only one processor. $\sum_{p=1}^{n} X_{ipk} = 1$

Thus the basic linear program is the following:

$$\begin{aligned}
& \text{Minimize } \sum_{p=1}^{m} \sum_{k=1}^{n} E_{ip} \\
& t_{ip} - t_{jp} \geq C_{ip} \eta_k X_{ipk} - M \alpha_{ij} \\
& t_{jp} - t_{ip} \geq C_{ip} \eta_k X_{ipk} - M(1 - \alpha_{ij}) \\
& t_{ip} + C_{ip} \eta_k X_{ipk} \leq d_i \\
& \sum_{p=1}^{m} X_{ipk} = 1 \\
& \sum_{k=1}^{n} X_{ipk} = 1 \\
& \sum_{p=1}^{m} \sum_{k=1}^{n} \sum_{i<j} \alpha_{ij} \geq 0 \\
& t_{ip} \geq 0 \forall i \\
& t_{ip} \geq r_i \forall i \\
& \eta_k \geq 0 \forall k
\end{aligned}$$

This model can be extended to adjust the deadlines whenever no solution exists even for the highest voltages.

$$\begin{aligned}
& \text{Minimize } x \\
& t_{ip} - t_{jp} \geq C_{ip} \eta_k X_{ipk} - M \alpha_{ij} \\
& t_{jp} - t_{ip} \geq C_{ip} \eta_k X_{ipk} - M(1 - \alpha_{ij}) \\
& t_{ip} + C_{ip} \eta_k X_{ipk} \leq x \cdot d_i \\
& \sum_{p=1}^{m} X_{ipk} = 1 \\
& \sum_{k=1}^{n} X_{ipk} = 1 \\
& \sum_{p=1}^{m} \sum_{k=1}^{n} \sum_{i<j} \alpha_{ij} \geq 0 \\
& t_{ip} \geq 0 \forall i \\
& t_{ip} \geq r_i \forall i \\
& \eta_k \geq 0 \forall k
\end{aligned}$$

Running example:
We continue with the system, in its stable state before applying any reconfiguration (Table 1, Section III). During execution, the system can undergo different reconfigurations through interactions with its environment and becomes infeasible. When applying our approach, different scenarios can occur.

**Scenario 1:** the system is feasible after the tuning of the scaling factors of the processor speeds.

The application of the proposed approach provides the optimal combination between the tasks and the processors in order to meet all timing constraints and optimize the energy and the makespan. For each task, we compute, as described previously, start time, finish time and new WCET after changing the scaling factors of the processor speeds. The results are presented in Table 3.

**Scenario 2:** the necessity of tuning the deadlines:
We assume now that a reconfiguration scenario has been applied and parameters of tasks have been updated. Results are shown in Table 4.
**Table 3: System behavior after reconfiguration**

<table>
<thead>
<tr>
<th>Task</th>
<th>CPU</th>
<th>Release time</th>
<th>Last WCE</th>
<th>Start time</th>
<th>Finishing time</th>
<th>Deadline</th>
<th>New WCE</th>
<th>New factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>P₁</td>
<td>0.00</td>
<td>20.00</td>
<td>0.00</td>
<td>4.00</td>
<td>70.00</td>
<td>8.00</td>
<td>0.40</td>
</tr>
<tr>
<td>T₂</td>
<td>P₁</td>
<td>0.00</td>
<td>22.00</td>
<td>8.00</td>
<td>16.80</td>
<td>80.00</td>
<td>8.00</td>
<td>0.40</td>
</tr>
<tr>
<td>T₃</td>
<td>P₁</td>
<td>0.00</td>
<td>30.00</td>
<td>12.50</td>
<td>30.00</td>
<td>90.00</td>
<td>25.00</td>
<td>0.60</td>
</tr>
<tr>
<td>T₄</td>
<td>P₁</td>
<td>0.00</td>
<td>28.00</td>
<td>0.00</td>
<td>11.20</td>
<td>110.00</td>
<td>11.20</td>
<td>0.20</td>
</tr>
<tr>
<td>T₅</td>
<td>P₁</td>
<td>0.00</td>
<td>32.00</td>
<td>0.00</td>
<td>12.80</td>
<td>100.00</td>
<td>12.80</td>
<td>0.40</td>
</tr>
<tr>
<td>T₆</td>
<td>P₁</td>
<td>0.00</td>
<td>50.00</td>
<td>0.00</td>
<td>66.20</td>
<td>85.00</td>
<td>30.00</td>
<td>0.60</td>
</tr>
<tr>
<td>T₇</td>
<td>P₁</td>
<td>0.00</td>
<td>65.00</td>
<td>16.80</td>
<td>68.80</td>
<td>94.00</td>
<td>13.00</td>
<td>0.80</td>
</tr>
<tr>
<td>T₈</td>
<td>P₁</td>
<td>0.00</td>
<td>80.00</td>
<td>11.20</td>
<td>91.20</td>
<td>105.00</td>
<td>80.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

**Table 4: Dynamic adaptation of deadlines and WCETs.**

<table>
<thead>
<tr>
<th>Task</th>
<th>CPU</th>
<th>WCE</th>
<th>Start time</th>
<th>Finishing time</th>
<th>Last deadline</th>
<th>New deadline</th>
<th>New WCE</th>
<th>Scaling factor</th>
</tr>
</thead>
<tbody>
<tr>
<td>T₁</td>
<td>P₁</td>
<td>78.00</td>
<td>55.60</td>
<td>86.60</td>
<td>86.00</td>
<td>94.54</td>
<td>31.20</td>
<td>0.40</td>
</tr>
<tr>
<td>T₂</td>
<td>P₁</td>
<td>68.00</td>
<td>00.00</td>
<td>27.20</td>
<td>78.00</td>
<td>82.72</td>
<td>27.20</td>
<td>0.40</td>
</tr>
<tr>
<td>T₃</td>
<td>P₁</td>
<td>80.00</td>
<td>00.00</td>
<td>35.00</td>
<td>99.00</td>
<td>106.36</td>
<td>35.60</td>
<td>0.40</td>
</tr>
<tr>
<td>T₄</td>
<td>P₁</td>
<td>108.00</td>
<td>80.80</td>
<td>130.00</td>
<td>110.00</td>
<td>130.00</td>
<td>43.20</td>
<td>0.40</td>
</tr>
<tr>
<td>T₅</td>
<td>P₁</td>
<td>98.00</td>
<td>78.98</td>
<td>118.18</td>
<td>100.00</td>
<td>118.18</td>
<td>39.20</td>
<td>0.40</td>
</tr>
<tr>
<td>T₆</td>
<td>P₁</td>
<td>84.00</td>
<td>00.00</td>
<td>33.60</td>
<td>85.00</td>
<td>100.45</td>
<td>33.60</td>
<td>0.40</td>
</tr>
<tr>
<td>T₇</td>
<td>P₁</td>
<td>92.00</td>
<td>74.29</td>
<td>111.09</td>
<td>94.00</td>
<td>111.09</td>
<td>36.80</td>
<td>0.40</td>
</tr>
<tr>
<td>T₈</td>
<td>P₁</td>
<td>104.00</td>
<td>35.60</td>
<td>77.20</td>
<td>105.00</td>
<td>124.99</td>
<td>41.60</td>
<td>0.40</td>
</tr>
<tr>
<td>T₉</td>
<td>P₁</td>
<td>71.00</td>
<td>27.20</td>
<td>55.00</td>
<td>74.00</td>
<td>87.45</td>
<td>28.40</td>
<td>0.40</td>
</tr>
<tr>
<td>T₁₀</td>
<td>P₁</td>
<td>83.00</td>
<td>33.60</td>
<td>66.80</td>
<td>85.00</td>
<td>100.45</td>
<td>33.20</td>
<td>0.40</td>
</tr>
</tbody>
</table>

**Simulated Annealing Approach**

Since the problem is NP-hard (10), it is a common idea to use a heuristic approach to achieve optimal solutions. The Simulated Annealing SA (26, 27) is based on neighborhood search. It starts with a random solution to improve it over iterations. Such heuristics always move from a solution to the best neighboring one. In order to escape local minima, SA allows different moves in a controlled manner. In each step, it generates a perturbation. If the objective function decreases, then the generated solution is accepted. Otherwise, the new state is accepted with a probability related to the increase.

- **Initial temperature** The temperature parameter plays an important role for accepting or rejecting objective functions. The initial temperature is fixed to 95. It determines the probability of deterioration,

- **Temperature length** The temperature length (40) is the number of iterations at a given temperature. However the temperature length may vary from temperature to temperature and is important to spend long time at lower temperatures,

- **Rate of temperature decrease** For less probability of accepting unfavorable solutions, the temperature should be decreased. The cooling ratio is the rate at which the temperature is reduced. In this paper, it is preferred to be fixed to ($\mu = 0.9$),

- **Stop criteria** In our simulation, the Simulated Annealing stops when the minimum value of the temperature reaches (5) or a certain number of iterations has been passed without improvement or when a number of 1000 iterations has been reached.

**NUMERICAL STUDY**

We have used ILOG CPLEX 11.1 solver to execute the integer programming model on a mono-processor core 2 duo, 1.2 Mhz and 1 Giga RAM. In our experimentation, we have randomly generated different task sets with 50 to 400 tasks. In Tables 5 and 6, the first column shows the size of the problem (number of tasks). The sub-column labeled "Time" indicates the computational time in milliseconds for each approach. The sub-column labeled "Energy" gives the total energy consumption. The sub-column labeled "Makespan" gives the maximum execution time from all tasks in the system.

**Comparison Between IP and SA**

Table 6 shows that the energy consumption of the applied integer program is lower than that of the heuristic. However for the large size instances, the heuristic is still much faster. We conclude that the integer programming is more efficient for the small instances. Moreover, the two approaches guarantee that all the constraints are respected. The following figures describe the evolution of the energy, the execution time and makespan for an instance of 200 tasks executed upon different multiprocessor platforms. According to the energy consumption, the Integer Programming (IP) is more effective than Simulated Annealing (SA) for almost different instances and platforms. In fact, it allows to explore more the search space of solutions and can give a fairly optimal solution. The SA proves its efficiency in terms of computational time and shows a large difference with the IP especially for large-size task sets and processors (Fig 2). About the makespan metric, the two approaches produce close results under a platform of 4 processors. More precise, when we multiply the number of processors, the IP approves its head advantage versus SA from

![Energy evolution with different number of processors.](image-url)
Table 5: Integer Programming Results.

<table>
<thead>
<tr>
<th></th>
<th>4 CPU</th>
<th>8 CPU</th>
<th>16 CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>801.21</td>
<td>868.15</td>
<td>311.14</td>
</tr>
<tr>
<td>100</td>
<td>1079.98</td>
<td>1391.45</td>
<td>479.62</td>
</tr>
<tr>
<td>200</td>
<td>1553.65</td>
<td>1689.55</td>
<td>665.20</td>
</tr>
<tr>
<td>300</td>
<td>1234.14</td>
<td>1899.85</td>
<td>788.43</td>
</tr>
<tr>
<td>400</td>
<td>1351.57</td>
<td>2137.49</td>
<td>919.39</td>
</tr>
</tbody>
</table>

Table 6: Simulated Annealing Results.

<table>
<thead>
<tr>
<th></th>
<th>4 CPU</th>
<th>8 CPU</th>
<th>16 CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>50</td>
<td>101.25</td>
<td>946.24</td>
<td>295.40</td>
</tr>
<tr>
<td>100</td>
<td>242.50</td>
<td>1389.56</td>
<td>485.60</td>
</tr>
<tr>
<td>200</td>
<td>422.96</td>
<td>1733</td>
<td>710.20</td>
</tr>
<tr>
<td>300</td>
<td>835.12</td>
<td>1945.55</td>
<td>855.68</td>
</tr>
<tr>
<td>400</td>
<td>1912.60</td>
<td>2388.71</td>
<td>956.10</td>
</tr>
</tbody>
</table>

Figure 2: Comparison between SA an IP based on computational time.

Figure 3: Comparison between SA an IP based on the Makespan.

Comparison with Related Works

We note that the approaches (IP and SA) provide more efficient results compared to those in (28, 25, 13, 29) according to a set of assessment metrics such as the energy, consumption, the execution time and the makespan. Our models allow to compute the scaling factors more than the execution sequence of tasks, the start and the finish time of each task. Table 7 shows that our approaches (IP and Heuristic) give better results than the works presented in (28) according to the computational time for instances of 10 to 25 tasks with a multiprocessor platform composed of 10 and 15 processors (Fig. 4). Knowing that the developed approach in (28) does not address the context of reconfigurable systems which undergo specific reactions and events during executions when communicating with the environment. To confirm the lower energy consumption, our approaches have been compared to works in (29) with an instance of 300 tasks (Table 8). The result shows that IP and SA are more efficient than HVSA and RH-VH VSA and give less energy consumption when the number of processors increases (Fig. 5). The work in (29) mainly focuses on the energy saving and does not take into account other constraints. Our solution combines more than a constraint such as the energy, makespan and computational time to produce a complete and balanced solution that respects different metrics and may be more appropriate for the reconfigurable systems. Our approaches, (13) and (25) are also compared in term of execution time and makespan (Table 9). The comparison confirms the originality and the performance of our works to give the optimal solutions either in the execution time (Fig. 6) or the makespan (Fig. 7).
Table 7: Comparison between IP, SA and Majazi (28)

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Processors</th>
<th>IP</th>
<th>SA</th>
<th>GA (28)</th>
<th>SA (28)</th>
<th>Lingo (28)</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>10</td>
<td>145</td>
<td>80</td>
<td>350</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>10</td>
<td>211</td>
<td>135</td>
<td>400</td>
<td>200</td>
<td>100</td>
</tr>
<tr>
<td>20</td>
<td>15</td>
<td>401</td>
<td>205</td>
<td>500</td>
<td>500</td>
<td>8500</td>
</tr>
<tr>
<td>25</td>
<td>15</td>
<td>803</td>
<td>460</td>
<td>900</td>
<td>800</td>
<td>81000</td>
</tr>
</tbody>
</table>

Table 8: Comparison between IP, SA and Chuan (29).

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Processors</th>
<th>IP</th>
<th>SA</th>
<th>HVSA (29)</th>
<th>RH-VHVS (29)</th>
</tr>
</thead>
<tbody>
<tr>
<td>300</td>
<td>8</td>
<td>1589</td>
<td>1645</td>
<td>1150</td>
<td>1080</td>
</tr>
<tr>
<td>300</td>
<td>16</td>
<td>1409</td>
<td>1384</td>
<td>1800</td>
<td>1720</td>
</tr>
</tbody>
</table>

Figure 5: Comparison between IP, SA and Majazi (28).

Figure 6: Comparison between IP, SA and Majazi (28).

Figure 7: Comparison between IP, SA and Majazi (28).

CONCLUSION

In this paper, we have considered an allocation and scheduling problem for periodic tasks upon multiprocessor platforms with DVFS capabilities. We have provided two efficient power aware approaches respectively based on integer programming and simulated annealing that preserve feasibility through appropriate task allocation. These methods provide online solutions for system reconfiguration under energy constraints. They act on processor speed and deadlines adjustment. The experimental studies confirm that our approaches mostly perform the schedulability requirements and effectively remain able to maintain the control and the stability of the system during execution. For future works, we intend to enrich the task model by considering additionally sporadic and aperiodic tasks. We also plan to implement the presented approaches as a real-time scheduling middleware.

REFERENCES


Table 9: Comparison between IP, SA, Chuan (29), Saricicek (13) and Sorafettin (25).

<table>
<thead>
<tr>
<th>Tasks</th>
<th>Processors</th>
<th>IP</th>
<th>MA</th>
<th>TS</th>
<th>SA</th>
<th>IP</th>
<th>MA</th>
<th>TS</th>
<th>SA</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>4</td>
<td>221</td>
<td>113</td>
<td>240</td>
<td>2670</td>
<td>188</td>
<td>57.2</td>
<td>67.43</td>
<td>75.05</td>
</tr>
<tr>
<td>20</td>
<td>8</td>
<td>266</td>
<td>156</td>
<td>2730</td>
<td>17390</td>
<td>1047</td>
<td>35.52</td>
<td>51.06</td>
<td>59.46</td>
</tr>
<tr>
<td>20</td>
<td>12</td>
<td>412</td>
<td>200</td>
<td>5690</td>
<td>17590</td>
<td>1078</td>
<td>15.38</td>
<td>29.17</td>
<td>34.62</td>
</tr>
</tbody>
</table>


Efficient Techniques for Test Generation and Validation of Reconfigurable Hardware Systems

Asma Ben Ahmed  
Tunisia Polytechnic School  
LISI Lab, INSAT  
University de Carthage, Tunisia  
E-mail: ben.ahmed.assma@gmail.com

Olfa Mosbah  
LISI Lab, INSAT  
University de Carthage, Tunisia  
E-mail: olfamosbah@gmail.com

Mohamed Khalgui  
SystemsControl Lab  
Xidian University, China.  
LISI Lab, INSAT  
University de Carthage, Tunisia  
E-mail: khalgui.mohamed@gmail.com

KEYWORDS  
Embedded System, Reconfiguration, Optimization, ATPG, Fault Detection, Validation, Simulation.

ABSTRACT

This paper deals with testing of Reconfigurable Hardware Systems (abbreviated, RHS) that should be adapted to their environment under well-defined conditions. RHS is considered as a network of components interconnected via signals and expressed in terms of data. A reconfiguration scenario is a run-time hardware operation allowing the addition/removal of hardware components to respond to changing requirements. The size of fault set is an important cost factor in the postproduction test of RHS. When comparing faults from different circuits, redundancy is possible. Consequently, inter-circuits relations do exist which lead to optimize the set of faults to be considered in the test generation. In this paper, we propose efficient techniques for test generation and validation of RHS. We present an automatic test generation algorithm for RHS based on the Multiple Target Fault Detection (MTFD). We aim to optimize the test generation time by generating test vectors for the minimal fault set with a high fault coverage. The proposed approach is complete since we propose a high level test generation which is able to provide the minimal test vectors that can be correctly and effectively used for testing. In order to prove the correctness of test generation process, parallel and fault simulations for RHS are presented. We propose a simulation environment named TriTest to implement the proposed solution which is applied to a STM32F4 board-based case study that we simulated to evaluate the papers contribution.

INTRODUCTION

Over the past few years, the realm of embedded systems has expanded to include a variety of products, ranging from home media systems, to smartphones, to embedded medical devices and sensor networks. Embedded systems are designed to perform specific tasks, often, with real-time computing constraints. In fact, these systems require stringent performance, energy efficiency and flexibility for multifunctional use. In order to cope effectively and timely with these requirements, reconfigurability has become an emerging approach in scientific research. In (Zhang et al. 2015), the authors define reconfigurability as the ability to flexibly modify system functions by adding/removing hardware/software components, modifying logic relation between components, or updating particular system data without any disturbance. Consequently, the reconfiguration can touch the software level (Gharbi et al. 2010), (Balani et al. 2006) and the underlying hardware (Ahmadinia 2007), (Ahmed et al. 2015).

Reconfigurable Hardware System (RHS) is a new manufacturing paradigm that aims to handle changes and provide the adequate functionality when it is needed. RHS allows postfabrication configurability, enabling single base hardware design to implement a variety of circuitries (Garcia et al. 2006). In our research, we are interested in the logical level design of RHS. We define, in (Ahmed et al. 2015), a RHS as a network of gates interconnected via signals and expressed in terms of data. A reconfiguration scenario is a run-time hardware operation allowing the addition/removal of hardware components.

The postproduction test is a highly important step in the production process of the RHS. This test is applied to each fabricated system in order to detect defective devices. It is carried out by checking the correct response of the hardware system under predefined input stimuli or test vectors (also called test patterns). A test vector for a particular fault is an assignment to the primary inputs of the circuit that leads to different output values depending on the presence of the fault. A core technique in this context is Automatic Test Pattern Generation (ATPG). The task of ATPG is to generate a test set that targets possible faults. A fault is an abstract model of a physical defect. Among the various existing fault models such that delay, opens and bridges fault model, the single stuck-at fault model is still the most popular because of its closeness to actual defects and the algorithmic possibilities it offers for
generating test vectors. Besides, it is shown that test vectors generated for single stuck-at faults are also effective for detecting other types of faults (Steininger 2000). Under this model, every single line can be permanently fixed (stuck) at a logical 1 (Line/1) or 0 (Line/0) value. The number of possible single stuck-at-fault is then given by \( 2^n \) where \( n \) is the number of lines in the device under test. In this paper, we assume a Single Stuck-at Fault (SSF) model.

The fault set size is a highly important factor in the post-production test: a large fault set leads to long test application time and exorbitant test costs. For RFS, the number of faults can be very large that can consequently lead to a significant slowdown of the test generation process using the ATPG tool (Schulz et al. 1988a) (Mahlstedt et al. 1990) (Higami et al. 2009). It is, thus, beneficial to minimize the number of faults whenever possible. Fault collapsing process reduces the number of faults to be considered in the test generation and the fault diagnosis. The approach presented in (Agrawal et al. 2003) (Veneris et al. 2004) proposes fault collapsing using classical Equivalence and Dominance relationships. These relationships can be said Intra-Equivalence and Intra-Dominance as well as they are interested in faults located at the same circuit. For RHS, when comparing faults from different circuits, redundancy is possible. Consequently, Inter-Circuits relations among faults do exist. In (Ahmed et al. 2015), we propose a new fault collapsing relationships termed Inter-Equivalence, Inter-Dominance and Redundancy reducing faults from different circuits of the RHS. This new classification of faults reduces considerably the number of faults which leads to optimize the set of test vectors generated using the ATPG process. Once the test set is made, parallel and/or deductive fault simulation are used to validate the ATPG process. The hardware test is finally performed: test vector is processed by the circuit and the circuit’s response is checked. This stimulus/response procedure is repeated for the different test patterns in order to detect possible faults.

In this research work, we are interested in automatic test generation and validation of test vectors for the RHS. We propose an enhanced test generation approach for RHS based on the Multiple Target Fault Detection that aims to reduce the test generation time by generating test vectors for the minimal fault set with a high fault coverage. In order to prove the correctness of the calculated test vectors, parallel and deductive simulation are used. The proposed approach is complete since we propose a high quality test set which is able to provide the minimal input stimuli that can be correctly and effectively used for testing. In contrast to previous methods, the novel method has the ability to reduce the test set size significantly. Finally, in order to verify and test our proposed solution, we developed a tool named TnTest and applied it to test a simulation of faults based on a STM32F4 board. Our experiments show that the size of the test set is reduced to just 8 efficient test patterns for an initial test set of 108 test vectors, when the Inter-circuits relations are considered.

The paper is structured as follows. The next Section is devoted to Reconfigurable Hardware System. In Section 3, the case study is described. Section 4 presents the Reconfigurable ATPG. Section 5 introduces the test vectors validation while Section 6 is devoted to experiment. Section 7, concludes this paper.

**RECONFIGURABLE HARDWARE SYSTEM**

**Formalization**

In our research, we are interested in RHS introduced in (Ahmed et al. 2015). RHS is modeled as a network of gates interconnected via signals and expressed in terms of data. A hardware reconfiguration will be any operation allowing the activation (addition) and deactivation (removal) of hardware components at run-time. Therefore, at a given time (t), a change may affect the gates, the signals or the input data. RHS is defined by the triple \( \alpha, \beta, \varphi \) as follows:

\[
RHS = (\alpha, \beta, \varphi)
\]

where \( \alpha = \{\text{components}\} \), \( \beta = \{\text{Signals}\} \) and \( \varphi = \{\text{Data}\} \). \( \alpha \) is the set of subsets of hardware components, \( \beta \) is the set of subsets of signals used to interconnect the different components and \( \varphi \) is the set of subsets of data used to express the system differently.

At a given time \( t \), the system is characterized as follows:

\[
RHS(t) = (\alpha(t), \beta(t), \varphi(t)) = (X, Y, Z)
\]

where \( X=\alpha(t) \) which denotes the set of components at time \( t \), \( Y=\beta(t) \) which denotes the set of signals at time \( t \), \( Z=\varphi(t) \) which denotes the set of input data at time \( t \). \( X \subset \alpha ; Y \subset \beta ; Z \subset \varphi \). As shown in (Ahmed et al. 2015), to address all possible forms of reconfiguration we define three levels: (i) **Architectural reconfiguration** triggered by creating, deleting or updating gates, (ii) **Structural reconfiguration** allowing activation/deactivation of internal signals between already used components, (iii) **Data reconfiguration** allowing selection of data via a multiplexer. This original classification covers all possible forms of reconstructions to dynamically change the behavior of any RHS depending on the modifications that can occur in its environment. In order to control the increasing complexity of the reconﬁgurable hardware design, an RHS is modeled with three levels of hierarchy: the top level is the Architecture, the second one is the Structure and the third level presents Data. To switch from a reconfiguration to another, an external multiplexer (E-MUX) is attached to the system such that multiplexer inputs present the different circuits implementing the different reconstructions. Assigning new values into the selector pins (Sel) reconfigures the system to implement a new circuit. Consequently, RHS\((t)\) is updated as follows:

\[
RHS(t) = (X, Y, Z, Sel - Comb(E - MUX))
\]

where \( Sel - Comb(E - MUX) \) presents the selector pins assignment enabling the activation of the corresponding circuit. RHS is characterized with \( \Omega \), which is the initial fault list that contains all possible single stuck-at faults.
Background

The fault list is a central part of hardware test of the RHS. One approach for considerably reducing complexity of the testing process is fault collapsing. Fault collapsing reduces the number of faults using Equivalence and Dominance relationships among faults located in one circuit (Agrawal et al. 2003) (Jha and Gupta 2003). In addition to existing relationships, we propose in (Ahmed et al. 2015) a new characterization of faults termed fault Inter-Equivalence, fault Inter-Dominance and Redundancy fault. This new classification of faults enables the reduction of faults located in different circuits of the RHS.

- **Definition 1: Fault Inter-Equivalence**

  Let $V^1_i$ and $V^1_j$ be two sets of test vectors that detect, respectively, two faults $f_i$ and $f_j$. Two faults $f_i$ and $f_j$ in different circuits $C_i$ and $C_j$ are said to be Inter-Equivalent if they are detected by exactly the same set of vectors $V^1_i = V^1_j$.

- **Definition 2: Fault Inter-Dominance**

  Let $f_i$ and $f_j$ be two faults in different circuits $C_i$ and $C_j$. Let $V^2_i$ be the set of all the test vectors that detect the fault $f_i$. Similarly, let $V^2_j$ be the set of the test vectors that detect the fault $f_j$. The Fault $f_i$ is said to Inter-Dominate the fault $f_j$ if $V^2_i$ contains $V^2_j$ that means $V^2_i \supseteq V^2_j$.

- **Definition 3: Fault Redundancy**

  A fault $f_i$ is called redundant in different circuits $C_i$ and $C_j$: (i) if it is located at the same primary input signal used by the both circuits. If the fault appears in the circuit $C_i$, then it will also appear at the same signal of $C_j$ (ii) if the fault is at an internal signal of the circuit $C_i$ and the same signal is present in $C_j$, then $f_i$ is said to be redundant in $C_i$ and $C_j$ when $f_i$ is detected by the same set of vectors.

Related works

The Automatic Test Pattern Generation (ATPG) is the task of calculating a set of test patterns for a given hardware system with respect to a fault model by the deterministic test pattern generator (Kirkland and Mercer 1988) (Kocan and Saab 2001). A test pattern for a particular fault is a combination of assignments of logic values to the primary inputs which simulate the fault (line justification) and monitor the given fault at least one of the primary outputs (fault propagation) (Schulz et al. 1988b). The hardware system is tested by applying a sequence $T$ of test vectors $t_1$, $t_2$, $t_n$ and by comparing the obtained output responses, $z(t_1)$, $z(t_2)$, $z(t_n)$. A test $t$ that detects a fault $f$ makes $z(t) = 0$ and $z_f(t) = 1$ or vice versa, where $z(t)$ and $z_f(t)$ are respectively the output of the fault-free circuit and the faulty one. Thus, calculating the Boolean difference of the faulty and fault free circuit yields all test patterns for a particular fault. The Boolean difference is a powerful method for determining test patterns because it is guaranteed to find a test pattern for a fault if there exists one (Wang et al. 2006). Therefore, the set of all tests that detect a fault $f$ is given by the solution of the equation:

$$z(x) \oplus z_f(x) = 1$$

where $z$ is the fault-free function and $z_f$ is the faulty function. The ATPG problem is known to be NP-hard (Venkatasubramanian and Agrawal 2014). Consequently, the generation of test vectors is a challenging problem that has been tackled by numerous solutions and algorithms. The best known algorithms are:

- **D algorithm** was the first true algorithm for ATPG (Roth 1966). The method that Roth proposes is based on the interconnection of D-cubes, a notation for the Difference calculus used to describe the difference between the good and the faulty circuit ($D$ and $D$). $D$ means binary one in the good circuit, zero in the faulty one $D$. Binary zero in the good circuit, one in the faulty one (Kocan and Saab 1999) (Kocan and Saab 2001).

- **Path-Oriented Decision Making (PODEM)** is an improvement over the D algorithm. PODEM was developed by Goel in 1981 (Goel 1981). The PODEM algorithm has succeeded in reducing the number of occurrences of backtracks in comparison to the D-algorithm (Karim et al. 2008).

- **Fan-Out Oriented (FAN)** algorithm is an improvement over PODEM (Fujisawa and Shimoson 1983). It limits the ATPG search space in order to reduce the number of occurrences of backtracks in the algorithm and to shorten the processing time between backtracks.

- **Methods based on Boolean satisfiability (SAT)** are sometimes used to generate test vectors (Balcarek et al. 2013).

One use of parallel and deductive fault simulation is to validate test vectors generated using the ATPG process. We describe both parallel and deductive fault simulation.

- **Parallel fault simulation** was proposed by Seshu and Freeman (Seshu 1965). Assuming that binary logic is used, one bit is sufficient to store the logic value of a signal. Thus, in a host computer using $W$-bit data word, each signal is associated with a data word of which $(W-1)$ bits are allocated for $(W-1)$ faulty circuits and the remaining bit is reserved for the fault-free circuit. $W$ is the number of bits in the data word on which bitwise operations can be performed in parallel. In each pass of parallel simulation, the fault-free circuit as well as $(W-1)$ faulty versions are simulated in parallel for a given test vector. If $\Omega$ faults have to be simulated for a test vector, then $\Omega/W$ passes are required. For each fault, an appropriate fault mask is used to inject the effect of the fault at its site. Stuck-at fault mask is comprised of two $W$-bit integers $M(S)$ and $F(S)$. $M(S)$ indicates the position of the bit modeling the fault: $M(S)_i$ contains 1 in the $i^{th}$ bit position of signal $S$ corresponding to the faulty value and 0s otherwise. $F(S)$ indicates
whether the fault is a stuck-at 0 or a stuck-at 1: \( F(S) \), contains 1 (0) in the \( i \)th bit position of signal \( S \) corresponding to the faulty value if the fault is a stuck-at 1(stuck-at 0) and 0 otherwise (Saab 1993). If any of the fault circuit versions being simulated in the pass correspond to a fault site at signal \( S \), then \( S \) is updated using the mask as follows:

\[
S' = S.M(S') + M(S).F(S)
\]

- **Deductive fault simulation** is based on logic reasoning rather than simulation (Armstrong 1972). For a given test vector, deductive simulation performs only the fault-free simulation and deduces, all at once, the faults that can be detected. In deductive fault simulation, a fault list \( (L_S) \) is associated with a signal \( S \). \( L_S \) is the set of faults that causes \( S \) to differ from its fault-free value. A deductive procedure is applied to all lines in a level-order from inputs to outputs. In this process, fault lists are generated for each signal. The fault list of a signal is derived from the fault lists at the inputs of the gate producing that signal and any faults associated with that gate. The process of deriving the fault list of a gate output from those of the gate inputs is called fault list propagation (Hahnov et al. 2003).

**Discussion and originality**

The main target of the industrial ATPG is a high fault coverage with a minimum number of test patterns. Numerous attempts have been made in creating and designing algorithms and efficient techniques to reduce the pattern count and, at the same time, yield a high fault coverage. The authors in (Messing et al. 2009) introduce two-dimensional fault list for ATPG. Different strategies for ordering the two dimensions have been proposed in order to reduce the number of generated patterns. The work in (Remersaro et al. 2009) propose a scalable method that relies on guidance for the ATPG decisions to generate close to minimal size test pattern. In (Baid and Srivastava 2013), the genetic algorithm proves to be an effective algorithm in finding optimum number of test patterns from the highly complex problem space.

As the new generation of embedded systems is addressing today new criteria as reconﬁgurability and flexibility, new techniques have to be developed that can cope with today’s circuits. Note that no one treated the problem of automatic test generation and validation of the RHS.

**CASE STUDY**

In this paper, we use the same case study proposed in (Ahmed et al. 2015) to expose our contribution and to be assumed in the following as a running example. RHS is designed for rapid adjustment of functionalities, in response to new circumstances by changing or rearranging the hardware structure. We present a case study where hardware components can be added, removed or updated as needed to implement different functionality as shown in Figure 1. Based on the reconfiguration scenarios introduced in (Ahmed et al. 2015), the RHS case study is composed of three main architectures (Arch1, Arch2, Arch3). Each architecture comprises three different structures Struct1, with \( 1 \leq i \leq 3 ; 1 \leq j \leq 3 \) . The different structures can be expressed in terms of \( S_1 \), \( S_2 \) and \( S_3 \) or in terms of \( S_4 \) depending on input selection values of the Internal MUX (I-MUX) present in each logic circuit in order to enable data reconfiguration. 16-to-1 MUX, which is considered as an External MUX (E-MUX), is added to our model to connect the desired circuit to the output and consequently express the hardware system differently. According to the case study, we present in the following, examples of the different forms of reconfiguration.

**Architectural reconfiguration**

Switching the value of selector inputs of the E-MUX from ‘0000’ to ‘0110’ enables the change from Arch1-Struct1 to Arch1-Struct3. This scenario represents an architectural reconfiguration occurred by the removal of gates AND and OR and the addition of gates NOR and NAND.

**Structural reconfiguration**

Consider Arch1. Switching the value of selector inputs of the E-MUX from ‘0000’ to ‘0001’ enables the change from Arch1-Struct11 to Arch1-Struct12. This scenario represents a structural reconfiguration: AND and OR gates are retimed while the interconnection between signals is updated. \( Z_{11} \) and \( Z_{12} \) are expressed differently as follows: \( Z_{11} = \left( (S_1.S_2) + S_3 \right).
S_4 + (S_1.S_2) \) \( S_4 \) \( Z_{12} = \left( (S_1.S_2) + S_2 \right).
S_4 + (S_1.S_3) + S_2 \). This scenario represents a data reconfiguration.

**Data reconfiguration**

Consider Arch1-Struct11. If \( X_1 = 0 \) then \( Z_{11} = (S_1.S_2) + S_3 \) else \( Z_{11} = S_4 \) where \( X_1 \) is the selector input of the I-MUX. According to the case study, examples of Inter-Equivalence, Inter-Dominance and Redundancy relations are given.

**Fault Inter-Equivalence**

We consider two faults \( S_2/1 \) and \( S_3/1 \) located, respectively, in Arch1-Struct11 and Arch1-Struct12. The sets of test vectors that detect \( S_2/1 \) and \( S_3/1 \) are as follows:

\[
V^{S_2/1} = \{1000, 1001\}
\]

\[
V^{S_3/1} = \{1000, 1001\}
\]

We notice that the two sets of vectors are identical. Therefore, \( S_2/1 \) is Inter-Equivalent to \( S_3/1 \).
Fault Inter-Dominance
Consider two faults $S_1/0$ and $S_1/1$ located, respectively, in $Archi\_Struct_{11}$ and $Archi\_Struct_{12}$. The sets of test vectors that detect $S_1/0$ and $S_1/1$ are as follows:

$V^{S_1/0} = \{0010, 0111, 0110, 0111\}$

$V^{S_1/1} = \{1010, 1011\}$

We notice that $V^{S_1/0} \subset V^{S_1/1}$. Therefore, The Fault $S_3/0$ Inter-Dominates $S_1/0$.

Fault Redundancy
Through the example given in Figure 1, $S_1$ is a primary signal used in $Archi\_Struct_{11}$ and $Archi\_Struct_{12}$. If a stuck-at-0 and a stuck-at-1 occur for $S_1$, then the faults $S_1/0$ and $S_1/1$ are considered redundant in $Archi\_Struct_{11}$ and $Archi\_Struct_{12}$.

The fault list $\Omega$ of the RHS case study is initially equal to 108 single stuck-at faults. As shown in (Ahmed et al. 2015), the new classification of faults reduces considerably the number of faults from 108 to only 8 faults which leads to optimize the set of test vectors to be considered in the automatic test generation of the RHS.

**CONTRIBUTION: RECONFIGURABLE ATPG (R-ATPG)**

**Basic Idea**

One central problem to testing the RHS is the determination of an optimal sequence of test patterns. Due to the limited storage capacities of automatic test equipment and long test application time, a large test patterns set signifies high test costs. In contrast to the existing algorithms D, PODEM and FAN which operate on all faults from the fault list, smaller fault set is integrated into the ATPG process. The main idea is to generate test vectors for the minimal fault set with a high fault coverage. The minimal fault set so-called Optimal Fault Set (OFS) is obtained using the Inter-Circuits fault collapsing introduced in (Ahmed et al. 2015). The OFS may contain faults located in different circuits modeling the RHS. These faults can figure in architectural, structural and data reconfiguration. In contrast to classical ATPG approaches which work directly on the whole system, R-ATPG target a subsystem presenting fault sites. In fact, a Sel-Comb, which presents the selector pins assignment of the E-MUX, is associated to each fault in order to enable the activation of the subsystem containing the fault sites. Switching the values of the Sel-Comb set enables the activation of the needed circuits. Consequently, for each fault in the OFS a test pattern is generated and applied to the corresponding circuit. The circuit’s response is then checked: if a fault is detected then the list of faults so called Target Fault Set (TFS) having a relationship with the detected fault is generated. This relationship can present an Intra/Inter Equivalence, an Intra/Inter Dominance and Redundancy relations. Therefore, test vectors that detect the faults of the OFS are guaranteed to detect all faults having a relationship with the fault and can consequently provide a high fault coverage. Thus, a test pattern for a given fault can target multiple faults simultaneously. This process is called Multiple Target Fault Detection (MTFD).

**Formalization**

In order to realize the proposed idea, we define a Dynamic Data Structure (DDS). The fault list OFS and the corresponding Sel-Comb set enabling the activation of the required subsystem are stored in DDS. A pointer to the TFS list is also associated to each fault of the OFS in order to get the overall fault coverage via Intra/Inter-Equivalence, Intra/Inter-Dominance and Redundancy relationships. Once the R-ATPG is processed, DDS is updated by adding the generated test vectors. Thus, the attributes of the DDS are organized as follow:

[Optimal Fault Set (OFS)] [Sel-Comb set] [pointer to the Target Fault Set (TFS)] [test set T].

**R-ATPG Algorithm**

The proposed R-ATPG engine is based on the D-algorithm. The different steps needed to perform a successful search of
test vectors are as follow:

- Initially, for a fault \( f \) in the OFS, activate the corresponding circuit presenting the fault site by assigning the corresponding Sel-Comb into the inputs selector of the E-MUX,

- Next, sensitize the fault \( f \) by finding a set of primary input values that causes the fault site in the good circuit to have a value opposite to the faulty value,

- Propagate the difference created by fault sensitizing to a primary output where we can observe it,

- Resolve conflict by carrying the effect of each assignment backward in the circuit. Record generated test vector in the test set \( T \),

- The above steps are repeated for all faults in the OFS.

**Algorithm I R-ATPG**

```plaintext
DDS : input/output data structure
While OFS ≠ 0
    Fault \( f = \) PopFault(OFS)
    ActivateCircuit(Sel-Comb(E-MUX))
    SensitizeFault(f)
    Propagate()
    Backtrace()
    Update(DDS) // add calculated test vector \( t \) to \( T \)
End While
return DDS
```

The test generation problem is known to be NP-hard (Venkatasubramanian and Agrawal 2014). The proposed algorithm attempts to reduce the complexity of the test generation process by optimizing the fault space considered by the R-ATPG process. R-ATPG provides the minimal test vectors that can be efficiently used for testing. R-ATPG is well suited to solve automatic test generation of RHS.

**Example**

Consider the RHS shown in Figure 1. It has a total of 108 single stuck-at faults that can be reduced to 63 faults if we use the classical Equivalence and Dominance relationships as shown in (Ahmed et al. 2015). However, considering the Inter-Circuits relations, which is the main topic in (Ahmed et al. 2015), reduces considerably the number of faults from 63 to 8 faults. Thus, the OFS consists of 8 faults located in different circuits as follow \( \{S_1/0(1), S_1/1(1), S_2/1(1), S_4/0(1), S_4/1(1), S_2/0(3), S_{11}/1(4), S_3/1(7)\} \) where \( S_1/0(1) \) presents \( S_1/0 \) located in Circuit \( id=1 \) as shown in Figure 1. If we suppose that for each fault of the corresponding fault list, we generate one test vector, test generation when considering the R-ATPG algorithm applies 8 vectors for an overall fault coverage. In contrast, test generation without the R-ATPG applies 108 or 63 test vectors, if we use the Intra-Circuit relationships. In the following, we present an application of the R-ATPG process: R-ATPG is applied to generate test vectors for \( S_1/0(1) \) located in \( Archi_{struct1} \) when \( X_1=0 \) as shown in Figure 2. Therefore, applying the test vector \('1100'\) to the Circuit \( id=1 \) and checking the circuit’s response allow the detection of the corresponding fault \( S_1/0(1) \) and all faults having an Intra/Inter Equivalence, an Intra/Inter Dominance and Redundancy relations with \( S_1/0(1) \) as shown in Table 1.

**Figure 2: An example of R-ATPG**

**Table 1: Target Fault Set**

<table>
<thead>
<tr>
<th>Detected Fault</th>
<th>Target Fault Set (TFS)</th>
</tr>
</thead>
<tbody>
<tr>
<td>( S_1/0(1) )</td>
<td>Detected Faults via Intra-Equivalence and Inter-Equivalence ( S_1/0(1), S_1/0(2), S_1/0(3), S_2/0(1), S_2/0(2) )</td>
</tr>
<tr>
<td>Detected Faults via Intra-Dominance and Inter-Dominance ( S_1/0(1), S_1/0(2), S_2/0(1), S_2/0(2) )</td>
<td></td>
</tr>
</tbody>
</table>

The different test vectors obtained using the R-ATPG process are shown in Table 2. We notice that, by using R-ATPG, the number of test vectors decreases drastically to 8 faults compared to the initial test set. R-ATPG clearly outperforms classical approaches.

**CONTRIBUTION: TEST VECTORS VALIDATION**

For RHS, parallel fault simulation and/or deductive fault simulation are required to prove the correctness of the R-ATPG process.

**Reconfigurable Parallel Fault Simulation (R-PFS)**

**Basic Idea**

For the RHS, the number of test vectors as well the number of faulty circuits to be processed in parallel can be very large which leads to a slowdown of the parallel simulation process. We propose a Reconfigurable Parallel Fault Simulation (R-PFS) which attempts to simulate the optimal faulty circuits.
for the optimal test set such that faults in the OFS will be simulated in parallel for the test set generated using the R-ATPG. Consequently, the data word size is efficiently reduced which lead to an optimization of the whole parallel processing. For a W-bit data word, W − 1 bits are allocated for faults in the OFS and the remaining bit is reserved to the fault-free circuit. The OFS consists of faults located in different circuits of the RHS. These faults can figure in architectural, structural and data reconfiguration. A Sel-Comb is associated to each test vector in order to determine the corresponding circuit where the given test vector have to be applied. A fault mask, which is comprised of the two W-bit integers $M(S)$ and $F(S)$, is associated to each signal $S$. As the computation of the fault mask depends only on the data word structure, a unified fault mask is generated for the triplet (Architecture-Structure-Data).

**Formalization**
The output data structure of R-ATPG (DDS) presents an input for the R-PFS. DDS is structured as follows: [Optimal Fault Set (OFS)] [Sel-Comb set] [pointer to the Target Fault Set (TFS)] [test set $T$].

**R-PFS Algorithm**
R-PFS consists of the following steps:

- In the first step, data word is generated using the faults obtained in the OFS while the most significant bit is reserved to the Fault-Free circuit ($FF$). If the number of faults $F$ is upper than the data word machine size, then divide $F$ into $g$ groups of $p$ faults.

- In the second step, for each test vector $t$ in $T$ activate the corresponding circuit by assigning the corresponding Sel-Comb into the inputs selector of the E-MUX.

- In the third step, inject $f_{1,p}$ faults: calculate $M(S)$ and $F(S)$ for corresponding signals ($S$).

- Next, for a test vector $t$, assign corresponding values at each primary input and simulate $p$ faulty circuits.

- Finally, compare results with the $FF$: a fault is detected if its bit value differs from that of the $FF$ bit.

```
Algorithm 2 R-PFS

DDS: R-ATPG result

$g$ = Divide($F, p$)

For each $g$ group of $p$ faults

GenerateDataWord()

For each $t$ in $T$

ActivateCircuit(Sel-Comb(E-MUX))

Inject($f_{1,p}$) //Calculate $M(S)$ and $F(S)$

AssignPrimaryInputs($t$)

SimulateFaultyCircuits($f_{1,p}$)

Compare(results, $FF$)

RecordDetectedFails()

End For

End For
```

**Example**
According to the case study, the data word size obtained using the traditional fault collapsing is 64 (63+1) bits. For the RHS, as the R-PFS is employed to faults obtained in the OFS, data word size is then 9 (8+1) bits as shown in Figure 3.

```
<table>
<thead>
<tr>
<th>Bit 1</th>
<th>Bit 2</th>
<th>Bit 3</th>
<th>Bit 4</th>
<th>Bit 5</th>
<th>Bit 6</th>
<th>Bit 7</th>
<th>Bit 8</th>
<th>Bit 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>FF</td>
<td>$S_{1/0}$</td>
<td>$S_{1/1}$</td>
<td>$S_{0/0}$</td>
<td>$S_{0/1}$</td>
<td>$S_{0/0}$</td>
<td>$S_{0/1}$</td>
<td>$S_{0/0}$</td>
<td>$S_{0/1}$</td>
</tr>
</tbody>
</table>
```

Figure 3: Data word representation for R-PFS

Figure 4 shows an example of parallel simulation applied to the test vector ’1100’. According to R-ATPG, test vector ’1100’ detects $S_{1/0}$ in $Arch1-Struct_{11}$ (Circuit id=1). The activation of circuit id= 1 is enabled via $Sel-Comb= 0000'$. For each signal $S$, $M(S)$ and $F(S)$ are given and the simulation of the Fault-Free circuit as the eight faulty circuits is performed. As shown in Figure 4, the second bit of the output $Z_{11} = 1011111111$, which corresponds to $S_{1/0}$ differs from that of the $FF$ bit. It is then proved that ’1100’ detects $S_{1/0}(1)$.

**Reconfigurable Deductive Fault Simulation (R-DFS)**

**Basic Idea**
In this subsection, we present the Reconfigurable Deductive fault Simulation (R-DFS). R-DFS can be faster than R-PFS
and well suited for complex systems. Unlike the classical deductive fault simulation, R-DFS benefits from the reduction in number of faults and therefore in test vectors to be considered in the fault simulation which leads to optimize the overall process. For each fault in the OFS, we deduce possible faults from the fault-free circuit values by the optimal test set generated using the R-ATPG process, in one single pass through the circuit structure. This concept is applied to the different circuits of the RHS where the test vectors have to be processed. The Sel-Comb set is used to locate the site of application of the given test vectors.

**Formalization**

The output of R-ATPG (DDS) presents an input for the deductive simulation. DDS is structured as follows: [Optimal Fault Set (OFS)] [Sel-Comb set] [pointer to the Target Fault Set (TFS)] [test set T].

**R-DFS Algorithm**

The different steps needed to implement an R-DFS are as follow:

- Initially, for a test in the Test set T of DDS activate the corresponding circuit by assigning the corresponding Sel-Comb into the inputs selector of the E-MUX,
- Next, simulate the fault-free circuit (FF) of the given test vector,
- Generate and propagate fault list LS through the circuit structure,
- Record detected faults,
- The above steps are repeated for all test vectors in the optimal test set.

**Example**

An illustrative example of deductive simulation applied to the test vector ‘1100’ is depicted in Figure 5: According to R-ATPG, test vector ‘1100’ detects S1/0 in Archii Struct II (Circuit id=1). The activation of circuit id= 1 is enabled via Sel – Comb= ‘0000’. The fault free circuit is simulated as shown in the Figure 5. Fault lists of each signal are generated: the fault S1/0 appears in LS1 because its presence causes the value of primary input S1 to deviate from its correct value of 1. We note that the faults that don’t appear in the OFS should not be included in fault lists. The fault list LS1 is then propagated to signals S5 and S4. We can conclude that the test vector ‘1100’ detects S1/0(1). Consequently, the test vector generated using the R-ATPG is correct.
vectors generated using the R-ATPG are checked. We notice that the results of both techniques are identical. Consequently, test vectors generated using the R-ATPG process are quite valid. In order to present a concrete application of the proposed solution, we perform a test experiment using TnTest and STM32F4DISCOVERY board (STMicroelectronics 2013) where RHS case study is simulated. A potentiometer is used to simulate defective system by inserting single stuck-at faults. A stuck-at 1 figures in signal S1 of circuit id=1. To test the proposed RHS, efficient test vectors already generated using TnTest are applied. Consequently, S1/1 is detected as shown in Figure 10. The TFS is then generated using TnTest as shown in Figure 11.

We should finally point out that no one in the related works treated the problem of hardware testing of the RHS. TnTest tool is then considered original and is the first one treating this problem.

Figure 6: Home Interface

Figure 7: R-ATPG results

Figure 8: R-PFS results

Figure 9: R-DFS results

Figure 10: Test experiment with STM32F4DISCOVERY

Figure 11: Scan results ’detected faults’

Evaluation

In order to illustrate the key benefits of our solution, we present a statistical comparison of the number of test vectors before and after applying the R-ATPG. Figure 12 depicts the

\footnote{We thank Ms Emma GUERFALA for collaboration to prepare this experimentation.}
test set as the number of gates modeling the RHS increases. Consider $T_B$ the test set before applying the R-ATPG using the classical approaches) and $T_A$ the test set after R-ATPG process. We notice that without using R-ATPG process, the number of test vectors is very high. In contrast, the test vectors after applying R-ATPG represented with the red line is efficiently reduced.

![Figure 12: Test set reduction using the R-ATPG](image)

Another resulting benefit of our contribution is the test time. We suppose that this time took 30 seconds per test vector. According to the test vectors generated as shown in Figure 12, we measure the total time which is defined as:

$T_{sec}$: number of test vectors,

$N_{sec}$: number of vectors,

Total test time ($TT$) = $\sum T_{sec} \times N_{sec}$.

Figure 13 shows the total test time as the number of gates increases. Let $TT_B$ be the test time before applying the efficient test vectors generated using the R-ATPG process and let $TT_A$ be the test time after applying the efficient test vectors. We notice that $TT_B$ requires large amounts of time. Consequently, testing is considered tedious and time consuming. In contrast, we notice that $TT_A$ demands less than quarter time needed to test the system before applying the efficient test vectors. If we suppose that 1 minute of test costs 1$, the total cost of a test system with 875 test vectors will be about 875$ without efficient test vectors compared to 111$ when considering efficient test vectors, we note that the difference between the two costs is considered tremendous in mass production.

**CONCLUSION AND FUTURE WORK**

The current paper presents an efficient R-ATPG based on the MTFD. In order to validate test vectors generated using the R-ATPG, R-PS and Ron (R-DFS) are introduced. In order to verify our proposed solutions, we developed a complete tool ($TnTest$) and applied it to test a simulation of faults based on a STM32F4 board. Unlike other test applications, $TnTest$ supports the RHS and performs a low cost test process thanks to the proposed algorithms. These algorithms can also be useful to generate test patterns to detect multiple stuck-at faults, open and bridging faults. In our future work, we plan to continue our research with the Optimal Built in Self Test (BIST) of Reconfigurable Hardware Embedded Components.

**REFERENCES**


Balcar J.; Fiser P.; and Schmidt J., 2013. Techniques for SAT-based constrained test pattern generation. Micropro-
cessors and Microsystems - Embedded Hardware Design, 37, 185–195.


HEALTH CARE SIMULATION
AN AGENT-BASED APPROACH TO REVEAL THE EFFECTS OF AGE-RELATED CONTACT PATTERNS ON EPIDEMIC SPREAD

Dietmar Neubacher
Nikolaus Furian
Siegfried Vössner
Graz University of Technology
Department of Engineering- and Business Informatics
Kopernikusgasse 24, 8010 Graz, Austria
email: dietmar.neubacher@tugraz.at

KEYWORDS
Agent-based Simulation, Complex Adaptive Systems, Epidemiology, Model Design, Networks

ABSTRACT
Contact tracing is a prevalent intervention strategy to control epidemic outbreaks. Simulating and predicting contacts among individuals can significantly improve risk assessment and support interventions. Past outbreaks highlighted the need to incorporate age-related mixing patterns in epidemiological models, as close contacts are crucial for the spreading. Several studies examined the effects of behavioral change and adoption of contact structures, but still few models apply these concepts. Simulations generally spread the disease along a defined network structure or local proximity.

The agent-based model introduced in this paper uses an implicitly generated network by stochastically generating contact events and assigning agents. This selection is based on activity factors and an initially generated contact probability matrix, which incorporates age-related mixing patterns and the distance between individuals. As agents modify their behavior based on their current state, the whole network structure can adapt over time. Various scenarios are simulated and diverse dynamics are observed, as age-related mixing patterns are incorporated. The proposed model traces contacts during simulations and can reveal epidemiological risks for diseases that are transmitted during person-to-person contacts. Results show that social mixing can significantly influence the spread of diseases and in the worst case accelerates an epidemic outbreak.

INTRODUCTION
Epidemiological simulation models are frequently used to investigate the spread of disease or to evaluate possible prevention strategies. Almost a century ago, the first deterministic models described infectious disease spreading with differential equations. These approaches separated the investigated population into several connected compartments, to represent agglomerations of individuals sharing the same health states. Three categories are commonly used: Susceptible (S), Infectious (I), and Recovered (R). In more detailed studies, various other compartments are added to illustrate partial steps in between; e.g., Exposed (E), which usually represents people that are not obviously infected.

The System Dynamics (SD) modeling and simulation paradigm is often used to represent this compartmental approach. Subsequently, researchers used SD to communicate various SEIR models in epidemiological science. Nevertheless, this compartmental approach assumes a homogeneous and perfectly mixed population, which is sometimes too restrictive for real systems (Rahmandad and Stemman 2008).

In recent incidents, contact tracing was a very effective intervention strategy to control an outbreak (WHO 2014). Thereby, all individuals with close contacts to the infected person have to be identified, isolated, and observed over a certain period of time. The exact same procedure can be represented in simulation models, which incorporate important social aspects such as age-related mixing patterns. Established micro-level approaches can simulate human behavior and the transmission of diseases on a very detailed level.

The individual reactions to epidemic outbreaks can influence the spread of infections, as people will adjust their common behavior. Funk et al. (2010) reported various modeling studies that incorporate behavioral change and the impact on the outbreaks. For instance, infected subjects may reduce their mobility, or even isolate themselves. In contrast, healthy individuals could also reduce or avoid contact to an obviously infected persons. Yet, such self-initiated behavior is a reaction to global or local information and strongly dependent on the social context. Hence, Epstein et al. (2008) suggest that behavioral adoption should be incorporated in infectious disease models. Their statement is based on a simulation model that considers fear as a trigger for individual behavior. In detail, agents can get sick, scared, or both at the same time. As an agent changes its state, a reaction is triggered. In the model from Epstein et al.
(2008) three possible types of agents behavior are classified: *fleers*, *hiders*, and *ignorers*. During simulations agents transmit fear and/or disease to their neighbors and act accordingly. Subsequently the dynamics of the spreading significantly changes and divergent patterns can be recognized.

Human interaction is crucial in epidemiological science, as infections often happen when people meet. In simulation models these person-to-person contacts commonly occur along initially determined networks (Gross et al. 2006, Rahmanad and Sterman 2008, Tian and Osgood 2011, Kassaie et al. 2013). These structures sometimes change or evolve over time, as relationships between agents are added or removed. This adaptive behavior affects the dynamics of the epidemic spread, as bistability and cycles can occur (Gross et al. 2006).

The model proposed in this paper generally assumes a fully connected and weighted network, as every agent has at least a small probability to take part in a contact. However, a fixed network structure is not explicitly defined, as the approach focuses on contacts and preferred participants. These links are sampled in course of the simulation and therefore the actual network structure is generated implicitly each simulated time step.

As agents interaction is not only subject to local proximity, individual behavior, rules and possible states need to be considered as well. Therefore microscopic data is needed, to describe agents at this very detailed level. A survey conducted by Mossong et al. (2008) recorded characteristics of contacts along various countries in Europe. The published data give a detailed insight into the behavior of humans and their mixing patterns. Drawing on the final statement in this survey, the aim of this paper is to use this microscopic data and enhance the accuracy of epidemiological models. Additionally, Iooss et al. (2010) used an agent-based model and socio-demographic data to build a virtual abstraction of Italy and observed age-related contact patterns. Both studies stated similar findings and lead to the assumption that the mixing behavior is an important aspect and can influence the spread of diseases. Finally, various age groups tend to have a higher ratio of physical contacts (Mossong et al. 2008). This aspect can be important as well, considering that virus transmission often depends on the type and duration of a contact.

Existing agent-based simulation models for disease spread use static and pre-defined networks, often based on local proximity. To break with this limiting assumption, the model proposed in this article dynamically generates contacts between individuals, taking individual motivation and attributes other than location, e.g. age related contact patterns, into account. Thereby, modeling flexibility is increased significantly as individual behavior is included. The authors are not aware of any previously published agent-based models that consider this kind of individual motivation for the generation of contact patterns among agents.

**MODEL DESIGN**

The Agent-based Modeling and Simulation (ABMS) paradigm is often used to analyze complex systems that are composed of many autonomously acting individuals within an environment (Macal and North 2010). Marshall and Galea (2015) state that ABMS is still a new technique in epidemiological science, but is obviously very useful to investigate stochastic person-to-person processes. Previous research compared compartmental approaches to agent-based models and stated a significant impact of heterogeneity and network topology on the dynamics of the outbreak (Rahmanad and Sterman 2008, Tian and Osgood 2011).

The agent-based paradigm is selected for various reasons. First, modeling autonomously acting individuals is the most realistic way to describe humans. Second, the available microscopic-data enables the possibility to model individual behavior on a more detailed level. Finally, the greatest advantage of ABMS is its flexibility (Bonabeau 2002). For example, the complexity of an agent is often depicted by a simple set of rules, which can be easily adjusted. The behavioral change can be individually modeled as a reaction to internal state changes or external events. Furthermore, the size of the investigated population can be easily adjusted by adding or removing agents.

In the following sections the main design elements of the agent-based model, underlying assumptions and the concept of agents interaction are explained. In the next step, the relevant attributes of individuals and the process of generating the set of contacts are proposed. Finally, the incorporated behavioral adjustments of agents is outlined.

**Population and Mixing**

According to Marshall and Galea (2015) a mathematical representation is chosen to illustrate the generic agent-based model. A population of agents $i \in \{1, ..., N\}$ is generated and each agent contains a set of $m \in \{1, ..., M\}$ attributes. Among the most important attributes are age, sex, location, health-state, and an activity factor. Finally, the population is stored as an $N$-by-$M$ matrix $A^t: \{1, ..., N\} \times \{1, ..., M\}, (i,j) \rightarrow a_{ij}$.

The agents ID is given by $i$ and $m$ represents the previously mentioned attributes. This matrix describes the state of the whole population at a specific point in time $t$ and is tracked throughout the simulation.

$$A^t = \begin{bmatrix} a_{i,1}^t & a_{i,2}^t & \cdots & a_{i,M}^t \\ a_{2,1}^t & a_{2,2}^t & \cdots & a_{2,M}^t \\ \vdots & \vdots & \ddots & \vdots \\ a_{N,1}^t & a_{N,2}^t & \cdots & a_{N,M}^t \end{bmatrix}$$

In addition, an $N$-by-$N$ Contact Probability Matrix (CPM) is used to store the possibility that two agents
meet \((c_{ij})\). Initially every agent has the same probability to be met by another agents. This value is weighted by several factors, which can be related to various important aspects, such as social behavior, location, individual preferences, and many more. As some of these aspects might not be of interest in specific cases, it is possible to consider only selected factors. The proposed model actually takes two aspects into account: the distance between agents and their age-related contact probability.

First, the factor \(c_{ij}^d\), is used to consider the distance between agents \((d_{ij})\) in an exponentially decreasing way.

\[
    c_{ij}^d = c_0^d \cdot e^{-\lambda d_{ij}}
\]  

(2)

Thereby, it is assumed that near agents will meet more likely. However, the model should also enable long distance contacts, because otherwise the simulation would often end in locally isolated outbreaks. Consequently, the probability that two agents meet decreases exponentially, as their separation increases. Note that the agents location attribute must not refer to a real geographical location. Brockmann and Helbing (2013) introduced the term effective distance and replaced conventional distances with probabilistically motivated ones. Their approach is very useful, especially in the alpine region, as some cities are separated by just a few kilometers, but are unreachable in a direct way. As geographic proximity does not correlate with the effort to reach certain locations, isochronous effective distances should be used. This representation illustrates the gap between agents as the time or effort to meet each other, instead of the real distance in a geographical sense.

The second factor \(c_{ij}^a\) represents an age-related contact probability. Following Mossong et al. (2008), mixing behaviors are quite similar in European countries, yet the amount and intensity of contacts differ. However, there are three impressive findings from Mossong et al. (2008), which urge to consider age-related contact patterns in epidemic simulation models. First, people tend to mix with others in the same age. This effect is most evident throughout childhood and teenage, when kids visit schools or kindergarten (see Figure 1). Second, another diagonal weight exists at approximately one generation (25-30 years) offset, representing the contacts with parents or children. Third, adults have more low-intense contacts to a broader age-set, as many of them occur during work. Additionally, this survey provides more detailed information about contacts in general, such as type, duration and occasion.

Finally, the entire CPM can be calculated.

\[
    c_{ij} = c_{ij}^d \cdot c_{ij}^a \quad i, j \in \{1, ..., N\}
\]  

(3)

As a result, individuals tend to contact others according to their age-related mixing pattern, but as the distance increases this preference declines. Further aspects, to capture gender-related or social mixing patterns, can be added if necessary. Note that the CPM is slightly bidirectional \((c_{ij} \neq c_{ji})\) as the age-related contact probabilities are dependent on the initiators. For instance, a 15 year old individual has a higher probability to initiate a contact to an infant, as in the opposite direction. Finally, self-contamination is often not relevant, therefore it is assumed: if \(i = j \rightarrow c_{ij} = 0\).

The CPM has to be recalculated if relevant attributes change. Nevertheless, age-related contact patterns will not shift without external influences. Furthermore, births and natural deaths are not considered, because they do not significantly modify the population during the observed period. Yet, disease related deaths and agents movement alter the CPM, but are primarily not incorporated.

**Heterogeneity**

During initialization each agent is assigned an age and sex among other attributes. Based on these values, every agent is also assigned a normally distributed initial number of daily contacts \(dc_i\) (based on Table 1). As, individuals will not have the same amount of contacts every day, the initially sampled \(dc_i\) will be used to determine the agents sociableness. Therefore, an activity factor \(f_i\) is calculated as follows:

\[
    f_i = \frac{dc_i}{N} \sum_{i=1}^{N} dc_i
\]  

(4)
$f_i$ is used to adjust the probability of being part of a contact in two ways. First, having a low activity factor means that one is less likely to be chosen as an active part of a contact. Second, it is assumed that a person having a lower activity factor is also more unlikely to be randomly met.

**Contacts**

The simulation model is executed using a periodical update once a day. Therefore, each simulated time step a new set of contacts will be sampled in the following procedure:

1. **generate contact**: Without loss of generality we model one-on-one contacts, so every contact involves two agents. Hence, in the first step the total number of possible contacts $TC$ is calculated.

   $$TC = \left[ \frac{1}{2} \cdot \sum_{i=1}^{N} dc_i \right]$$  \hspace{1cm} (5)

2. **sample contact initiator**: After generating this set of contacts, an agent will be assigned to each contact. Using the individuals activity factors, a Partial Distribution Function (PDF) is generated and the initiator $x$ is sampled. Therefore, the higher the activity factor of an agent, the more likely this agent will be selected.

   $$P_{(x=i)} = \frac{f_i}{\sum_{i=1}^{N} f_i}$$  \hspace{1cm} (6)

3. **sample contact partner**: Given the initial agent $i$ and the initially calculated CPM, all relevant contact probabilities are represented in $RC_i$. Yet, these probabilities are independent of the individuals sociability. To make sure that less active agents are also less likely to be selected, these values are multiplied by the corresponding activity factor.

   $$RC_i = \left\{ \begin{array}{c} c_{i,1} \cdot f_1 \\
   c_{i,2} \cdot f_2 \\
   \vdots \\
   c_{i,N} \cdot f_N 
   \end{array} \right\}$$  \hspace{1cm} (7)

Finally, another PDF generated with $RC_i$, is used to select the contact partner. As a result, preferred connections establish and agents can meet more than once a day. These preferences have a higher risk to spread a disease and are important to highlight in contact tracing interventions.

4. **sample contact type**: To conclude, every contact is either physical or non-physical. Using the data provided by Mossong et al. (2008) every age-mix has a specific probability to be physical (see also Figure 1).

The actual number of daily contacts per agent varies every time step and is not set to the initially sampled number. Furthermore, individuals are not restricted to meet only once a day. Applying these restrictions would lead to cumbersome contact patterns. In particular, the last contacts have to link the remaining agents, regardless of their location and age. Using the distribution functions to sample contact initiator and partner leads to a quit similar result, but does not enforce contacts. For this reason, the agents attribute daily contacts is used to define an activity factor, while the actual number of daily contacts is sampled each time step. Consequently, outlying active agents have a high possibility to be an initiator of a contact, but due to their separated location they may not be selected as a counterpart that often. By contrast, less active agents in agglomerated areas could be selected quite often as a partner, especially when they surrounded by many same-aged individuals.

**Adaptive Network**

Network topology plays an important part in agent-based simulations (Rahmandad and Sterman 2008). In fact, simulation models often use fully connected, random, small world, scale-free or lattice networks. In this approach, agents are linked together in stochastically generating contact events. The underlying assumption is a fully connected structure, because every agent has the chance to meet any other agent. As the actual network is generated each time step, it will adapt to changes during the simulation. This adaption can be based on several factors, such as individual behavioral change, prevention strategies, closure of transportation nodes, and many more. The proposed agent-based model implements behavioral change and the possibility to adapt the CPM. Both adjustments modify the implicitly generated contact structure over time and therefore change the dynamics of the epidemic spreading. For instance, agents individually react to situations and might break their link to others, so the network will adapt and outbreaks can end up in isolated circles (Gross et al. 2006).

**Behavioral Change**

The spread of diseases is strongly dependent on the individual behavior of the people (Gross et al. 2006, Epstein et al. 2008). Simulation models incorporate these reactions by adjusting various parameters. Such modifications can range from a state change of individuals, or an adjustment of model parameters, to a mutation of the entire contact structure (Funk et al. 2010). For instance, individual state changes are commonly used to investigate prevention strategies, like vaccination. The other modifications are behavioral changes as a consequence to the presence of a disease or individual beliefs about it. The classification is independent from the applied simulation paradigm. However, in contrast
to the frequently addressed compartment models, individual behavior is vastly better captured by the agent-based approach (Marshall and Galea 2015). Following the model from Epstein et al. (2008), individual reactions can change model parameters and subsequently the network structure. In the proposed model, the agents attributes are adjusted, with regard to the actual health state and subsequently the probability to participate in a contact. A healthy agent will be as active as initially defined. After being exposed to the disease, the agents health state changes to exposed and due to a discomfort the activity factor is assumed to decrease slightly. After a predefined incubation period the agents health state changes to infected and the activity factor will drop dramatically. This reduction does not change the CPM, but individual attributes and for this reason also the probability of being initiator or partner in a contact. Additionally, the number of individual daily contacts is reduced by the same proportion and consequently, the total number of contacts per day declines. So, the entire population shows a lower activity during an epidemic outbreak. This adaption also affects healthy individuals, as they will rarely be chosen to participate in a contact.

The CPM has to be modified in several situations, as agents might change their contact patterns due to an infection or an epidemic outbreak in general. In fact, prevention strategies like closing school and kindergarten significantly change age-related mixing patterns, as children and teens will reduce contacts with same-aged individuals. Moreover, if people are apparently sick, the ratio of physical contacts might also change, because of safer practices. Due to the flexibility of the agent-based approach, these assumptions can be easily implemented and added to the simulation model. For instance, the closure of schools can be represented by adjusting the age-related contact probability $c_{ij}$. Or, reducing the ratio of physical contacts to infected individuals can express safer practices. However, these assumptions have to be addressed when applied scenarios are examined. In the evaluated cases the spread of a disease without any prevention or vaccination strategies is simulated. The behavioral change is only incorporated as an individual activity reduction, related to their internal state of health.

**Diffusion**

Up to this state, the simulation model is completely independent from the specific disease and generates contacts only based on distance and age. The authors do not want to investigate a specific disease, rather reveal the arising effects as age-related contact patterns are incorporated. However, to examine these effects, a simplified non-lethal disease is added. It is assumed that the transmission can occur during physical and non-physical contacts. Yet, the probability of an infection is higher during physical link. Moreover, once recovered from the disease an agent will be treated as immune for the rest of the simulation run.

**IMPLEMENTATION**

In this section the implementation and initialized settings are stated. To achieve the highest flexibility the entire simulation model is built in a DotNet framework. Consequently, various initial settings and adjustments are possible. The following three scenarios are used to highlight the impacts on the dynamics of an outbreak, as age-related contact patterns are incorporated. First, the number of agents is determined at the beginning, but their initialized location can vary. The simulation model uses population density maps to distribute agents, to reveal the effects of age-related contact patterns when individuals are heterogeneously distributed. Second, the CPM is initialized with divergent incorporated aspects. Therefore, different settings can be analyzed with the same underlying simulation model. Third, the initial number and age-set of infected or exposed individuals is explicitly defined. In general, they are randomly selected from the population. However, as age-related contact patterns are incorporated, the age of an index case becomes more important. Therefore, the upper and lower age-limit for initially infected or exposed individuals can be set in the initialization phase. However, creating and handling the contact matrix will become extensive, as the number of individuals increases. The current model does not include any segmentation or other performance savings, so investigated
populations are limited to a few thousand agents on a standard computer.
Due to the stochastic characteristics, the following results were generated by evaluating $n = 50$ simulation runs each. To handle the computational effort, $N = 1000$ agents were used for the following observations. Furthermore, the population was situated in a continuous 500-by-250 environment. All scenarios use the actual demographic structure of Austria, which was gathered from Statistik Austria (2015). Other relevant initial settings are listed in Table 2.

**IMPLICATIONS**

**Initial Distribution of Agents**

Age-related contact preferences can strongly influence the patterns of epidemic spread. To reveal this impact the individuals are distributed in three different ways (see Figure 2). First, they can be homogeneously placed within the environment, to assume perfect mixing (2(a)). Second, agents are heterogeneously distributed but in a continuous manner, to avoid barriers (2(b)). In this case two agglomerations are determined: a dense area at the lower left side and another, but less dense, on the right side. In the third scenario, agents are distributed heterogeneously and discontinuously, due to an abandoned area between the denser areas (2(c)).

The results show that the total numbers of infections are quite similar for all initial distributions, but the dynamics are different. Due to behavioral changes, age-related contact patterns deviate and the impact of the location declines. Still, local outbreaks might occur, but agents do not care about their neighborhood so much. Due to the preference of same-aged individual, they overcome longer distances to contact a person, if necessary. Nevertheless, the amount of daily contacts per individual is independent from the environment. Consequently, locally isolated agents might be very active too, but all contacts will be with agents far away. In contrast, less active persons situated in denser areas might have a lower risk of getting infected, than remote active agents. These characteristics depend on $c_{ij}^{a}$, which declines exponentially, as the distance increases. To examine real scenarios, this function can be adapted to historical data.

To conclude, the distribution does effect the diffusion of the disease, but incorporating age-related mixing behavior reduces the impact. The amount of infections will not change significantly, because the number of daily contacts per individual is independent from the location. However, introducing denser areas or barriers will influence the propagation of the spread. As, denser areas significantly influence spreading based on local proximity, the following results are based on a homogeneous distributed population.

**Age-related mixing**

In this section, a distance based approach is compared to the introduced age- and distance-related method. Thereby, both approaches use the same simulation model, whereas the distance-based approach assumes $c_{ij}^{a} = 1$. In other words, the CPM does not consider age-related preferences. Furthermore, the probability for a physical contact is set to 65%, which is the mean value of all age-related probabilities. In both cases the age is sampled from the same socio-demographic data. As contacts to same-aged individuals happen most fre-
quent in childhood and teenage the results illustrated in Figure 3 are based on initially exposed agents between 10 and 20 years. Figure 3(a) shows the behavior when using the age- and distance-related CPM, whereas Figure 3(b) depicts the behavior when contact probability is solely based on the distance between agents. In both cases, the dynamics are similar, but there is a significant deviation, which is illustrated in Figure 3(c). The result indicates that age-related contact patterns can accelerate epidemic outbreaks. Furthermore, the number of new infections peak approximately 5 to 10 time steps earlier on a lower level.

This effect is achieved as the disease rapidly spread through specific age-bands and slower beyond. As a young exposed individual prefers to contact a same-aged agent even farther away, the disease can spread out more quickly. Local outbreaks might still occur, but the initially infected age-set tends to have more contacts as the rest. Subsequently isolated outbreaks merely occur in the early phase. The opposite behavior can be observed, if an elderly age-set is initially exposed. This age-group tends to have less contacts, so new infections will be rare when age-related contact patterns are incorporated.

### Index Case Dependency

In all previously analyzed scenarios the disease spreads from the initial infected age-set to any other. The duration until this occurs has drastic effects on the outcome of the simulation. It is already revealed that spreading in a young age-band can be much quicker and wider. Therefore it can be assumed that the potential for an epidemic outbreak is higher when the index case is situated in this age-set (Wallinga et al. 2006). Furthermore, another risk might occur, if the index case is exposed and infectious, but does not feel sick and sticks to its habits (WHO 2014). However, as agents do not change to their mixing behavior in this situation, this simulation approach can be used to quantify epidemic risks. In both cases, the age of initially infected individuals becomes important and age-related contact patterns have to be considered. Wallinga et al. (2006) have noted that models using age-related mixing can reproduce the spread of diseases like mumps and pandemic influenza better. Therefore, this section observes the dependency of the age of an index case and an epidemic outbreak, when age-related contact patterns are incorporated.

To highlight the effect, the simulation is adjusted as follows: First the model is initialized with a disease where transmission can solely occur during a physical contact (see Table 3). As younger people tend to have a higher ratio of these physical contacts with individuals of their own age-group, this modification amplifies the impact. Second, the exposed index cases \(E_{50} = 5\) are randomly chosen in a predefined age-set. To intensify the effect, the initially exposed agents will be between 10 and 20 years, as direct physical contacts frequently happen when these young individuals meet. However, infants and children have a high ratio of physical contacts with their parents. But initially exposing them will lead to a rapid infection of a mid-aged individual and does not lead to an epidemic spread within the same age-set. The reproduction ratio \(R_0\) represents the number of cases an individual generates during its infectious period. This value is often used to estimate the risk for an epidemic outbreak and serves as an indicator to reveal this dependency. \(R_0\) is generally related to a specific disease, and does not consider the age of the index case. Nevertheless, the results have shown significantly different \(R_0\), as age-related contact patterns (CP) are incorporated. Note that all simulation runs use the same initial disease. However, the model that generates contacts based on local proximity (NCP) indicates a low risk for an epidemic outbreak. By contrast, \(R_0\) is more

<table>
<thead>
<tr>
<th>Parameter Settings for a Disease with a Weak Physical Contact Transmission</th>
</tr>
</thead>
<tbody>
<tr>
<td>Infection Probability (%)</td>
</tr>
<tr>
<td>Exposed: non-physical</td>
</tr>
<tr>
<td>Exposed: physical</td>
</tr>
<tr>
<td>Infected: non-physical</td>
</tr>
<tr>
<td>Infected: physical</td>
</tr>
</tbody>
</table>

Figure 4: Dependency of the index cases (dotted lines indicate 95% CI). (a) higher \(R_0\) when age-related mixing are incorporated; (b) \(R_0\) dependent on the age of patient zero.
than twice as high in the CP approach (see Figure 4(a)). As a consequence, $R_0$ seems to depend on the age of the index case as well. To reveal this dependency various age-sets were initially infected and the results are illustrated in Figure 4(b). It can be observed, that the risk for epidemic outbreaks is highest when the index cases are in childhood and teenage. In contrast, elderly people tend to have also a high ratio of physical contacts, but the reproduction ratio is smaller. The reason for the slower spreading appears to be the smaller number of daily contacts and subsequently the low activity factor.

CONCLUSION

The direct person-to-person contact is commonly necessary to transmit diseases. While low intensive contacts might occur frequently by chance, as people encounter during daily business, close contacts are often grounded on individual behavior and preference. So, the single person becomes important and investigated populations should not be aggregated with a compartmental approach. Therefore, the introduced agent-based model incorporates age-related personal contact patterns and can reproduce disease propagation effects observed in real social networks.

Considering these age-related contact patterns is common in epidemic outbreaks, as tracing contacts is already a useful intervention strategy (WHO 2014). In comparison to most of the existing simulation models, where transmission is only based on near distance contacts, the proposed approach generates different behavior. It is observed that high risk age-groups spread the disease much faster over farther distances and accelerate the epidemic outbreak, as also noted by Wallinga et al. (2006). Furthermore, these high risk groups are very active and have a higher ratio of physical contacts simultaneously (Mossong et al. 2008). Our results indicate that considering contact patterns is crucial, since they can influence the behavior of the system.

At this stage we have used an established and validated mixing model in order to highlight the importance of segmenting the population to investigate contact based disease spread. In future research we will compare the performance of our approach with existing ones, using real epidemiological data.

REFERENCES


Jozzi F.; Trusiano F.; Chiniazzi M.; Billari F.C.; Zagheni E.; Merler S.; Ajelli M.; Del Fava E.; and Manfredi P., 2010. Little Italy: An Agent-Based Approach to the Estimation of Contact Patterns - Fitting Predicted Matrices to Serological Data. PLoS Computational Biology, 6, no. 12, e1001021.


Statistik Austria, 2015. Population by age and sex, as of 1.1. 2015.


TOWARDS A FRAMEWORK FOR HOLISTIC ANALYSIS OF HEALTHCARE SYSTEMS

Ignace Djitog  
African University of Science and Technology  
Abuja, Nigeria  
djitog2@gmail.com

Hamzat Olanrewaju Aliyu  
Federal University of Technology  
Minna, Nigeria  
hamzat.aliyu@futminna.edu.ng

Mamadou Kaba Traoré  
Université Blaise Pascal  
Clermont-Ferrand, France  
traore@isima.fr

KEYWORDS
Healthcare Systems, Modeling & Simulation, DEVS

ABSTRACT
We present a work in progress towards building a framework for holistic analysis of Healthcare Systems (HSs) through a disciplined stratification of concerns and a systematic integration of simulation processes in different layers. A lot of simulation-based research efforts can be found in the literature where HSs are studied with focus on perspectives such as the allocation of scarce health care human and infrastructural facilities to meet the needs of patients, disease spreading within an hypothetical community, and so on. The different perspectives are often studied in isolation with constant parameters as abstractions of the influences of other phenomena on the system under study. We propose a methodology for a "loosely" integrated simulation where independent simulation processes of disparate concerns in HS exchange live updates of their influences on one another. We think this approach will take the results obtained closer to the reality of the interactions between health phenomena and help stakeholders take more realistic decisions.

INTRODUCTION
HSs are complex systems of distributed subcomponents governed by complex health processes, inter-organizational workflow, and various services (Barjis 2011). Applications of Modeling and Simulation (M&S) to HSs usually target specific aspects of healthcare problems. Some of these studies include minimizing patients' waiting times in outpatient clinics (Mustafee et al. 2012; Topaloglu 2006), monitoring the flow of patients for efficient utilization of healthcare facilities (Mes and Bruens 2012; Morrice et al. 2013) and epidemiology researches to institute required policies in HSs (Kassie et al. 2013; Worth et al. 2010). Other areas include studies of human population and healthcare delivery (Charfeddine and Montreuil 2010) and HS management efficient use of scarce human and infrastructural resources for healthcare services delivery (Ma and Demeulemeester 2013; Harper 2002; Persson and Persson 2009).

Discrete Events Simulation (DES) methods have been used recently to study problems related to the performances of HSs (Gunal and Pidd 2010; Mes and Bruens 2012). Other methods combine simulation with optimization techniques, Data Envelopment Analysis (DEA), and goal programming, to study different aspects of healthcare problems (Ahmed and Alkhamsi 2009; Topaloglu 2006).

Interestingly, simulation processes to address different healthcare problems are often done in isolation. Since in reality, the system under study exists amidst other systems and phenomena that may influence its internal processes, a common approach to model such influences is to represent them as parameters in the model under study to experiment with different hypothetical values of the parameters in separate simulation runs. For instance, in a simulation of the allocation of healthcare resources to tackle the spread of a disease in an environment, the model may include some parameters as abstractions of the coefficients of the levels of infections, awareness, migration, etc. in the community. Then some hypothetical sets of values of the parameters are used for separate simulation runs to investigate the performance of the resource allocation. In reality, however, some (or all) of these coefficients could change within the periods of each simulation runs examined thereby making the modeler's assumption about them obsolete. Conversely, a simulation model of the epidemic itself may contain abstractions of coefficients mentioned previously including a parameter representing the level of healthcare resource allocations which are all maintained constant for different simulation runs of the epidemic model.

We argue that there is need to explore more pragmatic approaches to make situations, and hence, the results obtained as close as possible to reality. Therefore, we propose the parallel simulation of independent disparate simulation models of different problems whose outputs may influence one another and systematically transmit live updates and feedbacks between them. For instance, if we may simulate the epidemic model described previously concurrently with models that are sources of its parameters (i.e., levels of resource allocation, awareness, migration, etc.) and allow them to communicate new values of the parameters to one another. This approach will result in more accurate forecasts of the effects of the interactions between the different components of HSs and their responses to issues. We provide more details on the proposal in the next section, followed by a simple example as illustration; then we provide concluding remarks and directions for future work.

HEALTHCARE SYSTEM MODELING
Considering the complexity of the HSs and the diversity and requirements of simulation objectives in the domain, we propose a four-layered stratification of common M&S objectives for better placement of problems and selection of suitable formalisms to model the problems in a framework. Figure 1 shows the formulated layers in dashed boxes with double arrows representing the mutual interactions between them.
At layer A, the Spatial Allocation of Healthcare Resources represents resources (human and infrastructural) allocated to meet demands for healthcare services. Problems in this layer can usually be described as discrete events systems as it often involves scheduling of resources for specific services delivery and responses to service requests. A suitable formalism to model problems in this layer is the Discrete Events System Specification (DEVS) (Zeigler et al. 2000).

The Health Phenomenon Dynamics layer (layer B) represents the group of investigations of health-related phenomena in a community that may lead to a change in the demands for healthcare services. Examples of such phenomena include the disease spreading due to epidemics, seasonal occurrences, person-person contact, etc. Cellular Automata is usually used to model problems in this domain.

Layers C represents the category of problems to study how the dynamics in the population of a community may influence or be influenced by other health issues or the allocation of healthcare resources in the environment. Such dynamics in population may include birth, death, immigration and emigration rates. Such problems may be modeled with formalisms like Differential Equations. The individual behavior at level D describes the category of problems involving the investigations of human behaviors and personal habits such as educational level, physical state, emotion, cognition and social status in relation to other healthcare concerns and allocations of resources. Problems in this category may be modeled with Petri nets.

**INTEGRATION OF HS SIMULATION LAYERS**

In this section we describe the notion of “loose” integration between models in the different layers described in the previous section. Let us assume that Figure 2 represents the loose integration of simulation models, one in layer A and the other in layer C. We use the term “loose” integration to describe our notion of parameter integration here to indicate that the simulations models involved are not tightly coupled together as is usually the case in the couplings between the ports of models in the same layer; Rather, each model runs independently in its own experimental frame. Each model provides an input and an output interface similar to the update and notifier methods of the Observer for each of its input and output parameters. Therefore, when the simulations run concurrently, each output parameter notifies all its observers whenever there is a change in its value.

**CASE STUDY**

In this section, we report our experiment with the model by White (White et al. 2009) for simulating disease spread during epidemics. This study falls within layer B in our classification presented in Figure 1 and it is modeled with the two-dimensional Cellular Automata (CA).

Each cell of the CA is considered to represent a square area of the land in which the epidemic is propagating in a population and the state, $s_t^{a,b} \in [0,1]$, of any cell $(a,b)$ at any time, $t$, represents the ratio of infected population to the total population of the cell. Also, it is considered that the state of any cell, $(a,b)$, at any time $t$ depends on the states of its eight neighboring cells, $V' = \{(\alpha,\beta)|\alpha - 1 \leq \alpha \leq \alpha + 1, b - 1 \leq \beta \leq b + 1\} - \{(a,b)\}$ and that of the cell itself in the previous time step. They proposed a local transition function for each cell $(a,b)$ at a time step $(t + 1)$ as:

$$s_t^{(a,b)} = g\left(1 - P(t)s_t^{a,b}\right) + \sum_{a,b} \frac{\mu_{a,b}^{(a,b)}}{g} s_t^{a,b} + \sum_{\alpha,\beta} \mu_{\alpha,\beta}^{(a,b)} s_t^{\alpha,\beta}$$

Where: $P(t) = 0.2t + 0.2$ is a measure of the infected population that has recovered from the disease within the last time step, $g$ is a discretization function that returns a value in $[0,1]$, (see White et al. 2009) for more details. The real parameters $\epsilon$ and $\mu_{a,b}^{(a,b)}$ are characteristics of the epidemic and the environment with $\mu_{a,b}^{(a,b)} = c_{a,b}^{(a,b)} \times m_{a,b}^{(a,b)} \times \nu$ where $c_{a,b}^{(a,b)}$ and $m_{a,b}^{(a,b)}$ are abstractions of connections/links and movement of infected people respectively between cell $(a,b)$ and its neighboring cells $V'$ and $\nu$ is the virulence of the epidemic. The authors presented a simulation of the model with the following constant assumptions of the parameters: $\epsilon = 0.4$, $c_{a,b}^{(a,b)} = 1$ indicating that there exist connections/transportation links between every cell and its neighbors; $m_{a,b}^{(a,b)} = 0.4$ representing the movement of infected people between the different cells and $\nu = 0.4$.

**Our Proposal**

We claim that instead of constant parameters as abstractions of other healthcare concerns in the present simulation, it would be reasonable to have real models of the situations running concurrently and exchanging feedbacks with the disease spread simulation. We believe that the results obtained from this approach will give more accurate representation of reality. For instance, in reality, an outbreak
within a cell can lead to change in the pattern of movements between the cell and its neighboring cells; a model of this situation can find its place in layer B of Figure 1. It is also noteworthy to mention that this model of the movement pattern may be influenced by other factors such as the activities of healthcare facilities in the cell under study and its neighboring cells (models in layer A of Figure 1) and individual habits and characteristics such as awareness, immunity or vaccinations (layer D models).

We cannot provide elaborate models of all factors in this short paper due to space limitations; we have, however, repeated the same simulation twice for the same number of time steps. In the first simulations, we maintained the constant values proposed by the authors (i.e., White et al. 2009) and in the second, we used randomly generated numbers between 0 and 1 as the values of $m^{(a,b)}$ (the movement of people between the cells) which were assumed to be coming from an independent simulation running concurrently. We present the results in the next subsection.

**Simulation Results**

Recall that the state of any cell $(a, b)$ represents the ratio of the infected population to the total population which is a real number within the range $[0.0, 1.0]$. For simplicity, the values are rounded to one place of decimal to have a finite state set, $S = \{0.0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0\}$. The color code chosen to represent the states of the cells in successive time steps are given in Table 1:

<table>
<thead>
<tr>
<th>States</th>
<th>.0</th>
<th>.1</th>
<th>.2</th>
<th>.3</th>
<th>.4</th>
<th>.5</th>
<th>.6</th>
<th>.7</th>
<th>.8</th>
<th>.9</th>
<th>1.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Colors</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The simulation was done with an hypothetical space divided into 200 by 200 cells. Figure 3 and Figure 4 show the spread of the disease within the same group of cells when all parameters are fixed and when the parameter representing population dynamics was varied respectively. Each of the two figures shows the levels of spread in nine successive time steps. We see from Figure 3 that with fixed parameters, the disease spread appears to subside in successive time steps from step 3 with most cells having infection rates of below 30% at the ninth time step. However, with a varied population dynamic parameter in Figure 4, the rate of disappearance of the disease is slower and the infection rates within most cells are still above 50%.

For further comparison of our results, we observed the evolution of the epidemic within a selected cell at the centre of the space under study. The graphs of the infection rates against time steps for fixed and varied population dynamics parameter are provided in Figure 5 and Figure 6 respectively. The graphs also show significant difference in the evolution of the disease within this cell.
CONCLUSIONS
In this short paper, we have presented an ongoing research towards a framework for integrated simulation of different aspects of healthcare systems with exchange of live updates as influences between independent simulation processes. We proposed disciplined stratification of healthcare concerns that are often investigated with simulation into four categories and a systematic integration of the simulation processes in the four categories by mutual exchange of live parameter updates. Our approach is different from the state-of-the-art in that different healthcare concerns are usually studied in isolation while other health phenomena that may affect them are represented by some fixed-valued parameters. We believe that such influences are often not fixed in reality as the different components of Healthcare systems are expected to respond to other concerns and adjust some internal activities. In future research, we intend to develop the proposed loose integration mechanism into a framework to coordinate the exchange of influences among the studies such that every simulation process publishes changes in its variables that may be required by other processes in real time without disrupting their activities. Such framework can be used to establish a liaison between existing healthcare simulation environments for holistic analysis of healthcare systems.

REFERENCES


SIMULATING THE IMPACT OF OPTIMIZED DISPATCHING STRATEGIES FOR PATIENT TRANSITS

Nikolaus Furian
Michael O'Sullivan
Cameron Walker
Simon Riezboz
University of Auckland
Department of Engineering Science
Email: nfur003@aucklanduni.ac.nz

Dietmar Neubacher
Siegfried Vössner
Graz University of Technology
Department of Engineering and Business Informatics
Email: dietmar.neubacher@tugraz.at

KEYWORDS
Health Care, Transportation, Discrete Event Simulation

ABSTRACT

The on-time completion of patient transits can be identified as a bottleneck for the efficiency of health care services in major New Zealand hospitals. Delayed transits of patients between wards and treatment or diagnostic facilities lead to increasing waiting times at clinics and inefficient resource utilization (e.g., surgery teams) as appointment times are not met. Therefore, a long-term research project was launched to install optimized and automated dispatching algorithms for staff members performing those transits. In this paper, we briefly outline a stochastic simulation model that was designed to evaluate the improvement gained by optimized dispatching and present a set of algorithms for the allocation of transits to orderlies. Initial experimental results on several real-world instances are presented and discussed.

INTRODUCTION

In this article, we present the first steps and results towards the installation of automated optimal dispatching strategies in major New Zealand hospitals, characterized by multiple wards and treatment facilities. During their stay, patients frequently require transportation to and from appointments within the hospital (a transit). These transits are usually performed by an orderly. However, patients with additional needs, due to their condition, require the assistance of a nurse. Transits are required to arrive at the destination location within a certain time-window, usually between 15 minutes prior the appointment and the appointment time. However, the actual system performance deviates significantly from this target for two reasons: transit requests are made with not enough notice by the corresponding facilities; the dispatching of transits to available resources (orderlies and nurses) is done in an ad hoc manner. While the first issue may only be overcome by applying and enforcing stricter policies for transit requests, which is not within the scope of this article, the second issue may be tackled by the installation of automated optimal dispatching strategies, as described in this paper. Therefore, an optimization problem is defined and an algorithm to find an optimal, or at least near-optimal, solution is proposed. To evaluate the effectiveness of the algorithm in a dynamic and stochastic environment, a simulation model is implemented for evaluation.

The paper is outlined as follows: in the next section a brief description of the optimization problem is given. In the following sections the simulation model and the proposed algorithms are presented. The paper concludes with experimental results, concluding remarks and suggestions for further research.

PROBLEM DEFINITION

The problem can be described as follows. For a given point in time there is a set of known transits to be performed in the future. They include three different activities: preparation; the actual travel; and a handover activity. Each transit is defined by pickup and drop-off locations, staff requirements, and two associated time windows. The first window denotes the desired completion time of the transit, including the handover activity, and is usually chosen as $[−15, 0]$ with respect to the appointment time (note, all times are in minutes). The second time window covers delayed and penalized completion times that are still feasible, usually chosen to be $[0, 15]$ with respect to the appointment time. Note that early arrivals, e.g., $(−∞, −15)$ to the appointment time rarely occur in the real system but should be avoided by any automated dispatching routine.

Per definition, for transits that require an orderly and a nurse, the nurse is responsible for the transit, i.e., performing preparation and handover tasks. Consequently, orderlies only participate in the actual travel for this type of transit. Furthermore, there are given staff rosters that determine the availability of orderlies and nurses at any point in time. The goal, of any dispatching strategy, is to assign transits to staff members.
so that the overall lateness of all pending transits is minimal, i.e. the sum of all late completion times minus the respective appointment times. This problem is closely related to vehicle routing problem with semi soft time windows (VRPSSTW), as optimal routes through a network of jobs for orderlies and nurses have to be computed. However, the authors are not aware of a previously published problem instance including the need for synchronized routes of multiple staff types. For more detailed information and algorithms on VRPSSTW, the reader is referred to (Qureshi et al. 2010). A general description of routing problems with time windows can be found in (Kallehauge 2008). Note that staff members start and finish their shifts at a base, to which they also return in the case of no pending job. As the set of known (or logged) and pending transits to be performed constantly changes (at the start of jobs and arrival of new ones) re-optimization of current solutions have to occur on a regular basis. To allow the evaluation of optimization strategies within such a dynamic environment, a simulation model has been implemented.

THE SIMULATION MODEL

Simulation, and in particular Discrete Event Simulation (DES) is a commonly used tool for analysis and decision support in the health care industry. For reviews the reader is referred to (J.B. et al. 1999, Brailsford et al. 2009, Güinal and Pidd 2010). The main purpose of the implemented simulation model presented here is to allow the evaluation of automated and optimized dispatching strategies within the changing environment of the actual system. This includes: available information on future transits at any given point in time; and the stochastic nature of the real system (random travel, preparation, and hand-over times). This means that the optimized schedule of task allocation will deviate from the real system as time progresses. To address this issue, we conceptualized a DES model according to the Hierarchical Control Conceptual Modeling (HCCM) framework, see (Furian et al. 2015). The main feature of HCCM models are that the model control policies (e.g. triggering of conditional activities and/or dispatching decisions) are centralized in control units, rather than being nested in de-centralized process definitions or activity conditions. This methodology perfectly suits the underlying problem where dispatching is done by a centralized control using automated algorithms.

In the remainder of this section the basic components of the simulation model are briefly outlined. First, the underlying movement model is introduced as it is the basis for all activities that include travelling of staff members, with or without patients, between any two locations in the model. Second, individual behavioral paths of patients and staff members are described. Based on these paths and corresponding requests the control structures of the model are defined. Finally, some notes on the model implementation are made.

Movement Model

The first step of the design of the simulation model is the generation of a network consisting of all facilities within Auckland City Hospital, as well as paths and elevators between different locations. Based on this network a movement model of staff with and without patients is created (some paths may not be used when traveling with patients). In particular, when dispatched, a staff member calculates the shortest path from her/his current location to the destination using Dijkstra's algorithm and starts the movement. The travel time for the move is computed based on the total non-vertical distance to be traversed and the number of floors that have to be passed. Different velocities for movements with or without patients determine the time required to move along non-vertical paths. Note, that we assume that all staff members travel at the same speeds. The usage of elevators to change floors is simplified by sampling an exponentially distributed waiting time for the elevator and constant times for each floor to be passed.

Behavior of Staff Members and Patients

As previously mentioned, a transit request consists of three activities to be performed: preparation; the transits itself; and a handover activity. The generation of transit requests is based on historical transit data. Each request includes the following data: the log time (historic time when the request enters the system); start and end location; staff requirements; appointment time; and historical dispatch time. Note that preparation and handover durations are sampled from exponential distributions, whereas the time required to perform a transit is a result of the movement model described in the previous section. Hence, the process of a patient can be summarized by “Arrival - Wait for Dispatch Time - Wait for Staff - Preparation - Wait for Additional Staff (if required) - Transit - Handover”. The dispatch time of transits is computed based on expected durations of activities associated with the transit and denotes the time when staff members are dispatched to perform tasks (note staff members may need to complete current tasks first).

The behavioral path of staff members is not as simple as for patients, since in the case of multi-staff transits, preparation and handover is performed by the responsible nurse only. Further, when there are no transits pending in the near future, idle staff members return to the base location. As this results in decisions with higher complexity to be made (due to multiple staff potentially being involved in multiple tasks across the hospital), this logic is located in a designated and centralized control unit. Figure 1 shows the process for staff members (or-
Model Control Structure

As described above, the model control mechanism that handles requests for transits is nested in a centralized control unit. It includes a set of rules that reflects the decision logic of the model. The rule set is evaluated each time the system changes, which occurs in DES models upon the triggering of events. Events occurring in this model are: start and end events of activities; arrival events of patients; and events representing the dispatch time of transit requests. The control logic can be summarized as follows:

1. If a new transit has arrived or been completed dispatching decisions are updated by calling the optimization algorithm;

2. For each idle staff member:
   (a) check if she/he has jobs pending;
   (b) If no jobs are in the personal queue and the staff member is located outside the base, then she/he is sent to the base;
   (c) If there are existing jobs in the personal queue, then
       i. For transits that require immediate dispatching of the staff member, she/he is sent to that location;
       ii. If there is not enough time to return to the base, then the staff member is sent to the pick-up location of the next job and waits;
       iii. If there is enough time before the next job to return to base, then the staff member is sent to the base;
   (d) When located or arriving at the pickup location it is checked if the staff member is responsible for the transit: if responsible the preparation of the transit is triggered; if not, the staff member is sent in a “Waiting for Other Staff” activity;

3. For each transit with a completed preparation it is checked if all required staff members are present at the pickup location; if so the travel is launched.

Note, with regards to dispatching and the update of staff positions: non-idle staff members are assigned an earliest available time based on expected remaining durations of their current transit as an input for the dispatching algorithm; the optimization is allowed to entirely change previous assignments, if the transits have not started; staff members that are waiting for their next jobs to become due may be re-assigned to other transits by the dispatching algorithm; non-responsible orderlies that wait for the nurse to complete the preparation activity are not re-assigned to other jobs.

Model Implementation

The model implementation was done using an object-oriented and activity based DES library, previously developed by the authors in the C# programming language. Its main functionality is closely coupled with the principles of the HCCM framework, that is centralizing model execution logic in designated control units. In particular it consists of two event triggering mechanisms: first, a scheduled event list, that is standard to almost any DES library; second, conditional behavior (events and activities) that is only triggered within the proposed control units according to the rule sets specified by the user. Hence, the usual DES scanning of activity or process activation conditions is replaced by the execution of centralized rule sets, as described by Furian et al. (2014b).

OPTIMIZATION ALGORITHMS FOR TRANSIT DISPATCHING

In this section an optimization method for the optimal, or near optimal, transit dispatching problem is proposed. It consists of several sub-algorithms of increasing complexity, that are called sequentially if the previous one failed to find an optimal solution. Therefore, it has to be noted that the maximum of unavoidable delays for
all transits considered individually, which itself is given by the delay that occurs when sending the set of staff members that are able to complete the job earliest, is a valid lower bound. In many cases this denotes a tight lower bound which reduces the use of more runtime-intensive algorithms. Next, the proposed algorithms are briefly outlined individually, followed by a description of their combined use embedded in the simulation.

Next Possible Heuristic

The Next Possible Heuristic (NPH) is not only the simplest algorithm presented, but also approximately describes the ad-hoc policy that is currently in place to dispatch transits in the real system. It loops through the set of transits, ordered with respect to appointment times, and assigns staff members that can start the job earliest (based on expected availability times).

Enumerative Search Algorithm

The Enumerative Search Algorithm (ESA) makes use of the fast and simple NPH algorithm. As explained above NPH loops through the ordered set of transits (with respect to appointment time) for assignment. ESA breaks this ordering and defines its search space by a sub-set of all possible sequences of jobs. The subset is defined in such a way that the violation of ordering by appointment time is only allowed to some extent (i.e. jobs may only be switched in the sequence if their appointment times deviate not more than 5 minutes). Based on that search space a set of sequences is enumerated and each sequence is evaluated with the same assignment strategy as NPH. The best solution found is returned.

Combining Optimal Individual Solutions

The Combining Optimal Individual Solutions (COIS) algorithm makes use of the observation that the share of transits that require the assistance of nurses is rather small for practical instances. Hence, it optimally solves the problem reduced to orderlies and corresponding transits only. Therefore, a Mixed Integer Program (MIP) was formulated that represents the orderly only problem. This is then solved using a typical column generation approach for the VRPSSTW extended by some problem-specific enhancements. However, the exact description of the methodology is beyond the scope of this paper.

Based on the optimal solution for orderlies COIS attempts to fit nurse routes to the given allocation schedule. Therefore, critical time windows for nurses are computed. Basically, they denote the maximum delay of jobs with respect to the completion times given by the orderly solution without causing additional delays. If it is possible to find a solution with zero delay for the nurse problem with respect to the updated time windows, both solutions can be combined to an optimal overall dispatching decision. If it is not possible, either a feasible solution has been found (which may not be optimal) or no solution could be generated by the COIS algorithm. In that case, the best solution found by other algorithms is used as a basis for dispatching decisions.

Insert Job to Path

The last algorithm presented, Insert Job to Path (IJTP), is a simple heuristic that attempts to insert additional jobs to the routes found at the solution of the last call of the optimization method.

Optimization Framework

As mentioned above, the outlined algorithms are used in sequential manner to make dispatching decisions within the simulation model. Starting with the attempt to make use of the previously found solution (checks of potential for improvement in case of removed jobs due to stochasticity and IJTP), NPH, ESA and COIS are called after checks if the currently best solution is optimal, as illustrated by figure 2. The resulting overall algorithm is denoted by Optimization Framework (OptFram). Note that it may occur that no feasible solution with respect to hard time windows could be found by any of the algorithms in OptFram. In the real system, this transit would be still performed and the infeasibility would be accepted. To allow for this behavior we enlarge the hard time windows in the case that ESA was not able to find a feasible solution. This decision was motivated by the observation that the runtime of COIS increases dramatically if an empty set of feasible starting solutions is provided.

RESULTS

In this section we outline the process of simulation fitting and provide experimental results for a selected set of instances based on data collected at the Auckland City Hospital.

Simulation Fitting

Data available on historic transits includes locations, staff requirements, appointment time, dispatching time and performing staff members. However, we do not have any knowledge on the decision when to dispatch tasks and at what location staff members were when they received their dispatch order or respectively started a job. The lack of this information, combined with the inaccurate nature of the available data led to the following approach: First, traveling speeds of staff members, preparation and handover times, as well as elevator waiting times and traveling speeds were fitted on a filtered set of transits.
Experimental tests have shown that the following values for parameters lead to a Pareto-minimum over the mean deviation of transit times and the standard deviation: traveling speeds (with and without patients) of (50, 70) [m/minute]; exponentially distributed preparation, handover and elevator waiting times with means (1, 2, 0.25) [minute]; and elevator floor-change times of 0.5 seconds.

Second, we compared the historic overall system performance of a large data set with the simulated performance using only NPH as a dispatching routine, as shown by figure 3. As results were satisfactory we do not need to evaluate OptFram on small instances of historic dispatching decisions data that most likely include high uncertainty of quality, but are able to compare its performance with delays observed using NPH. Note that, for this analysis early arrivals were allowed (as we used historic dispatching times), even if they must be avoided by any future automatic dispatching policy.

Experimental Results

To evaluate the performance of OptFram we created 10 instances based on weekday transit data with corresponding appointment times between 8:00am and

8:00pm. The average number of transits performed during this period was 443.3 out of which on average 47.0 required a nurse to be present. It was assumed that 4 orderlies and 2 nurses were present the whole period and an additional orderly and nurse were present between 9:00am and 6:00pm. This denotes a realistic, but rather low, staffing level. Each instance was simulated 20 times and the average and worst performance of both each mode (NPH only and OptFram) was reported. Note that the same historic arrival streams were used with the exact same activity durations for each mode to ensure comparability. Table 1 shows the collected results, where \( t_i \) denotes the aggregate lateness of all transits, \( l_i \) the number of transits completed outside the desired time window and \( i_{ln} \) the number of transits completed outside the hard time window. One can clearly see that the OptFram yields a reduction in the average and maximum overall lateness and number of late arrivals over all simulation runs for all but one instance.

Table 2 shows performance measures of OptFram, including: the average instance size of optimization calls \( \varnothing \); the maximum instance size of a single optimization call \( \max \); the average share of calls that resulted in an optimal solution \( \varnothing_{opt} \); the average runtime of optimization calls \( \varnothing_{rt} \); and the average runtime of the most complex COIS algorithm \( \varnothing_{rt,cou} \). Note that a three minute time limit was applied to COIS. In case this limit was reached the solution obtained by ESA is used.

Results show that optimality is reached in a vast majority of cases and computation times are within a range that allows the usage of OptFram in the real world operating system where decisions must be obtained within minutes due to the constant re-planning.

CONCLUSION AND FURTHER RESEARCH

In this paper we proposed a simulation model to reproduce patient transits within major New Zealand hospitals. Further, we briefly outlined a optimization methodology for dispatching of staff members to transits. The algorithm was integrated into the simulation model and its performance compared to an ad-hoc heuristic that represents the policy currently used. Results show that significant reductions of overall lateness
<table>
<thead>
<tr>
<th>NPH</th>
<th>( \phi_{lt} )</th>
<th>max ( l_t )</th>
<th>( \phi(l_n, l_{nt}) )</th>
<th>max ( l_n, l_{nt} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mo-I</td>
<td>154.61</td>
<td>221.23</td>
<td>(27.6,2.7)</td>
<td>(35.4)</td>
</tr>
<tr>
<td>Tu-I</td>
<td>402.75</td>
<td>598.75</td>
<td>(73.65,2.5)</td>
<td>(100.7)</td>
</tr>
<tr>
<td>We-I</td>
<td>348.13</td>
<td>641.47</td>
<td>(81.94,2.11)</td>
<td>(118.5)</td>
</tr>
<tr>
<td>Th-I</td>
<td>184.64</td>
<td>213.9</td>
<td>(36.1,1.85)</td>
<td>(43.2)</td>
</tr>
<tr>
<td>Fr-I</td>
<td>307.77</td>
<td>523.66</td>
<td>(44.58,4.11)</td>
<td>(63.10)</td>
</tr>
<tr>
<td>Mo-II</td>
<td>130.4</td>
<td>197.95</td>
<td>(28.9,0.3)</td>
<td>(40.1)</td>
</tr>
<tr>
<td>Tu-II</td>
<td>297.86</td>
<td>376.89</td>
<td>(40.5,3.35)</td>
<td>(44.6)</td>
</tr>
<tr>
<td>We-II</td>
<td>205.35</td>
<td>327.2</td>
<td>(42.45,6)</td>
<td>(58.3)</td>
</tr>
<tr>
<td>Th-II</td>
<td>141.86</td>
<td>179.76</td>
<td>(23.85,1.35)</td>
<td>(33.2)</td>
</tr>
<tr>
<td>Fr-II</td>
<td>553.81</td>
<td>800.17</td>
<td>(84.75,5.7)</td>
<td>(103.15)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>OptFram</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mo-I</td>
</tr>
<tr>
<td>Tu-I</td>
</tr>
<tr>
<td>We-I</td>
</tr>
<tr>
<td>Th-I</td>
</tr>
<tr>
<td>Fr-I</td>
</tr>
<tr>
<td>Mo-II</td>
</tr>
<tr>
<td>Tu-II</td>
</tr>
<tr>
<td>We-II</td>
</tr>
<tr>
<td>Th-II</td>
</tr>
<tr>
<td>Fr-II</td>
</tr>
</tbody>
</table>

Table 1: Simulation-Optimization Results

| Inst. | \( |J| \) | \( n \) | max \( n \) | \( \phi_{opt} \) | \( \phi_{rt} \) | \( \phi_{rt_{cons}} \) |
|-------|-----|-----|--------|-----------|-------------|----------------|
| Mo-I  | 457 | 15.09 | 32    | 0.96      | 2.47        | 1.89           |
| Tu-I  | 466 | 14.78 | 33    | 0.86      | 2.47        | 16.73          |
| We-I  | 397 | 11.47 | 22    | 0.96      | 0.66        | 7.99           |
| Th-I  | 408 | 12.18 | 22    | 0.97      | 1.17        | 8.85           |
| Fr-I  | 444 | 13.64 | 23    | 0.97      | 1.08        | 9.79           |
| Mo-II | 451 | 12.37 | 26    | 0.97      | 0.59        | 4.3            |
| Tu-II | 430 | 15.94 | 32    | 0.96      | 5.04        | 5.88           |
| We-II | 447 | 11.7  | 21    | 0.92      | 1.43        | 10.12          |
| Th-II | 491 | 15.86 | 28    | 0.92      | 2.47        | 3.63           |
| Fr-II | 452 | 13.49 | 26    | 0.96      | 2.19        | 8.39           |

Table 2: Optimization Performance Measurements, times in seconds

and number of patients arriving late can be achieved by the usage of the optimization algorithm. As these are initial results, it is left for further research to analyze a broader set of instances and multiple staffing rosters. Furthermore, a comparison of results achieved under stochastic and deterministic environment may be of interest.

ACKNOWLEDGEMENTS

The authors would like to thank Tim Winston and Jonathan Wallace for their inputs and cooperation, as well as Thomas Adams for the basis of floor-plan mod-

els of Auckland City Hospital. Further, the first author thanks the Austrian Science Fund (FWF): Project Nr. J3376-G11, for funding his research in New Zealand.

REFERENCES


ENVIRONMENTAL SIMULATION
COMPARING NEURAL NETWORKS AND PSEUDO-INVERSE MATRIX TO FLOOD FORECASTING OF THE AMAZON RIVER
Marcio Menezes
Marly Costa
Cicero Costa Filho
Technological and Information Center
Federal University of Amazonas
Av. General Rodrigo Otavio Jordão Ramos, 3000
Aleixo, Manaus, AM, Brazil. 69077-000
E-mail: cffcilho@gmail.com

KEYWORDS
feedforward neural networks, forecasting tools, pseudo inverse matrix, seasonable rivers

ABSTRACT
Forecasting the maximum peak flood of rivers with seasonable behavior is an important task from an economic and social point of view. In this paper, we compare the performance of two forecasting methods for the peak flood of the Amazon River: feedforward neural networks and pseudo inverse. To improve the generalization of feedforward neural networks, the regularization training stopping criteria was employed. The neural network performance was calculated as a mean value of 100 simulations. For both forecasting methods, Pacific Ocean climatic indices and the river level measured at the port of Manaus are used as input variables. The best forecasting coefficient value was obtained with pseudo inverse network, \( r_f = 0.756 \).

INTRODUCTION
Some important watersheds, such as those of the Amazon, Congo and Nile rivers, are characterized by seasonal yearly floods. Forecasting the maximum annual flood level of one of these rivers is a difficult task, since the precipitation-runoff relationships are among the most complex hydrological phenomena to understand, due to the huge spatial and temporal variability of watershed characteristics, precipitation, as well as a number of characteristics that must be taken into account in modeling the physical processes. In this paper we are concerned with forecasting the peak flood levels (maximum annual water level) of the Amazon River. The variations that occur annually in the Amazon River have a tremendous impact on economic activities (fishery, agriculture and logging) that takes place in floodplains along sediment-loaded, nutrient-rich white-water rivers (várzea). According to Junk et al. (2000), these regions have the highest human population densities in rural Amazonia due to abundant natural resources, including highly-productive nutrient-rich soils, with easy accessibility. With an annual discharge of 6300Km³, the Amazon River is the largest river in the world in terms of the basin size. The peak flood usually occurs in June and the lower water level, in October or November. Figure 1 shows a histogram of maximum and minimum water level in the period of 1903-2013. The river level has been measured since 1903 by the Manaus Port of Authority (Manaus Port of Authority Port 2015). Different methods were used in the literature for forecasting hydrologic phenomena. Table 1 shows a brief summary of studies published in recent years concerned with these phenomena.

The forecasting applications shown in Table 1 varies from hours to months. The main tools used for forecasting are: feedforward neural networks (FNN), RBF neural networks (RBFNN), support vector machines (SVM), statistical correlations and genetic program. The input variables used for forecasting are of different types: previous rainfall and streamflow, water level in previous moments, the Southern Pacific oscillation index, Tropical Pacific Sea Surface Temperature, daily discharge time series and seasonal discharge time series.

Two studies listed in Table 1 address the issue of forecasting the peak flood of The Amazon River. Schongart and Junk (2007), employing multiple correlation, and analyzing the period of 1903-2004, obtained a high correlation \( r=0.71 \), \( p<0.0001 \), between the peak flood of The Amazon River and the South Oscillation Index of the South Pacific Ocean. Uvo et al. (2000), for the period 1946-1992, used FNN to forecast the Amazon River discharge. The training-test method was a 4-cross validation one. The FNN architecture was 8-20-1. The best performance of the neural network \( r=0.86 \) was obtained using the following input variables: Sea Surface Temperature (SST) index of the equatorial Pacific sea and the rainfall of three regions downstream of the city of Manaus.

In this paper, we compare the performance of two forecasting methods for the peak flood of The Amazon River: FNN and Pseudo-Inverse (PI) methods. To improve the generalization of FNN, the regularization training stopping criteria was employed. Pacific Ocean climatic indices and the river level measured at the port of Manaus were used as input variables for both methods.

MATERIALS AND METHODS

Materials
The input variable set of the forecasting methods is comprised of Pacific Ocean climatic indices and of the river level in months prior to the month of maximum water level.
The Pacific Ocean climatic indices are shown in Table 2: SST El Niño_{1+2}, SST El Niño_{3}, SST El Niño_{3+4}, SST El Niño, and the Southern Oscillation Index (SOI) of El Nino South Oscillation (ENSO) (Trenberth and Stepaniak 2001). The SST indices are defined as the mean temperature of the West Pacific Ocean in the regions shown in Table 2. The SOI index is calculated as the standard difference between the atmospheric pressure in the sea level measured in Tahiti (17°S 149°W) and Darwin, Australia (13°S 131°E). These index data were obtained for the period 1951-2014 (Climatic Prediction Center 2015). Table 3 shows the values of these climatic indices.

<table>
<thead>
<tr>
<th>Reference</th>
<th>Application</th>
<th>Forecasting</th>
<th>Tools</th>
<th>Input variables</th>
<th>Performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Chang 2001</td>
<td>Three layers RBF Neural Network</td>
<td>Rainfall and the streamflow information for up to 3 h before the forecast</td>
<td>Correlation coefficients between 0.817 and 0.999 at l=1, l=2 and l=3.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Raslan 2013</td>
<td>Three layers RBF Neural Network and Inverse Model</td>
<td>Water level in previous moments</td>
<td>Root mean square error of 0.038m in a 10 minutes forecasting from 06/03/2011 to 11/03/2011.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Schongart and Junk 2007</td>
<td>Water level 4 months before the maximum water level that occurs in the second half of June</td>
<td>Statistical correlation</td>
<td>Pacific South Oscillation Index (SOI) and water level at February.</td>
<td>A correlation coefficient of 0.71, p&lt;0.00001 in the period of 1903-2004.</td>
<td></td>
</tr>
<tr>
<td>Hu 2011</td>
<td>Support Vector Machine</td>
<td>Antecedent runoff at the observed station, past rainfall, and the average rainfall on the intervening area.</td>
<td>Runoff relatively error of 0.126.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pang 2011</td>
<td>Artificial Neural Network (ANN), Linear Perturbation Model (LPM), and Artificial Neural Network with Non-Linear Perturbation Model (ANN-NLPM).</td>
<td>Seasonal discharge time series between 1998 and 2003.</td>
<td>Nash-Sutcliffe model efficiency index of 91.13 for ANN-NLPM, 88.43 for ANN and 84.84 for LPM.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ENSO indice</th>
<th>Latitude</th>
<th>Longitude</th>
<th>Period</th>
</tr>
</thead>
<tbody>
<tr>
<td>SST El Niño_{1+2}</td>
<td>0° a 10°S</td>
<td>90° a 80°W</td>
<td>1950-2014</td>
</tr>
<tr>
<td>SST El Niño_{3}</td>
<td>5°N a 5°S</td>
<td>150° a 90°W</td>
<td>1950-2014</td>
</tr>
<tr>
<td>SST El Niño_{3+4}</td>
<td>5°N a 5°S</td>
<td>170° a 120°W</td>
<td>1950-2014</td>
</tr>
<tr>
<td>SST El Niño_{4}</td>
<td>5°N a 5°S</td>
<td>160°E a 150°W</td>
<td>1950-2014</td>
</tr>
</tbody>
</table>

SOI Standard pressure difference at sea level between Tahiti (17°S a 149°W) and Darwin, Australia (13°S a 131°E)
Table 3: Input variables and statistical significance

<table>
<thead>
<tr>
<th>Indices / River level</th>
<th>Measures</th>
<th>Months used for forecasting</th>
<th>Months not used for forecasting</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Jul</td>
<td>Aug</td>
</tr>
<tr>
<td>El Niño 1-3</td>
<td>r</td>
<td>-0.109</td>
<td>-0.166</td>
</tr>
<tr>
<td>t - Student Significant for p&lt;0.001?</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>SST El Niño 3</td>
<td>r</td>
<td>-0.280</td>
<td>-0.269</td>
</tr>
<tr>
<td>t - Student Significant for p&lt;0.001?</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>SST El Niño 4</td>
<td>r</td>
<td>-0.371</td>
<td>-0.359</td>
</tr>
<tr>
<td>t - Student Significant for p&lt;0.001?</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>SST El Niño 5</td>
<td>r</td>
<td>-0.260</td>
<td>-0.260</td>
</tr>
<tr>
<td>t - Student Significant for p&lt;0.001?</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>SOI</td>
<td>r</td>
<td>0.321</td>
<td>0.243</td>
</tr>
<tr>
<td>t - Student Significant for p&lt;0.001?</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
<tr>
<td>River level</td>
<td>r</td>
<td>0.158</td>
<td>0.122</td>
</tr>
<tr>
<td>t - Student Significant for p&lt;0.001?</td>
<td>N</td>
<td>N</td>
<td>N</td>
</tr>
</tbody>
</table>

Figure 1: (a) Occurrence frequency of The Amazon River flood maximum level in function of the month. (b) occurrence frequency of The Amazon River flood minimum level in function of the month.

The first value (left column) is the measurement made in July of the previous year (usually one month after the occurrence of the previous year’s peak flood). The last value (right column) is the measurement made in June of the year that the forecasting was done.

The river level in months prior to the month of maximum water level are also shown in Table 3. As explained previously for the climatic indices, the first river level corresponds to the value in July of the previous year (usually one month after the occurrence of the previous year’s peak flood). The last river level corresponds to the value in June of the year that the forecasting was done.

Table 3 shows these variables, their correlations with the maximum flood level, the value of t-Student statistical significance level and if this test is not significant at 0.1% level. As shown in Table 3, SST El Niño1+3 does not have a significant correlation with the maximum flood level. SST El Niño2+4 in December has the high correlation with the maximum flood level, r = -0.428. The SST indices present a negative correlation, while the SOI index presents a positive correlation. In this study the peak flood forecasting was calculated in February, such that only data collected until February are used for prediction. The same procedure was adopted by Schongart and Junk (2007).

Methods

Feedforward Neural Networks

The algorithms employed for forecasting the peak flood of The Amazon River is like the leave-one-out method of training and testing (Sonka and Fitzpatrick 2000). In the leave-one-out method, the FNN is trained with N-I patterns and tested with pattern 1. This process is repeated N times and a mean performance value is obtained. In algorithm 1, the only difference is that for calculating the peak river flood, m neural networks, with randomized weights and bias initializations, are trained and used to m forecasting of the peak flood. A mean forecasting value is then calculated. This method takes into account that the neural network behavior depends on the initialization sets of weights and bias. In this study m = 100 and N = 64 (1951-2014) were used. The algorithm is listed as follows:

1. Define an input variable set;
2. Select statistically significant n variables;
3. For each year (year_k)
a. Train $m$ neural networks with data of $N-1$ years ($N$ years - $year_x$). For each trained FNN, calculate a forecast value for the peak flood, $p_f$.
b. With the $m$ forecast values of $year$ calculate a mean value: $f_k = \frac{\sum_{i=1}^{m} p_f}{m}$.

In this study, the number of simulations $m = 100$ and the number of $n$ variables varied from 1 to 9. These variables are chosen from those shown in Table 3 as the ones with highest correlation values with the maximum flood level. The architecture of FNN used was s-s-1. The best value of $s$ was determined for each value of $n$. The back propagation algorithm was associated with the optimization method of Levenberg-Marquardt (Cichoki and Unbehauen 1993). The neuron activation function was the hyperbolic tangent. To improve the neural network generalization, the regularization training stop criteria was employed (Doan and Liong 2004). The objective of regularization is work with lower values of FNN weights and bias, resulting in a FNN with a smoother behavior. With this goal, the mean quadratic error criteria usually employed for FNN training stop is modified to the one shown in equation (1).

$$\text{msereg} = \gamma . \text{mse} + (1 - \gamma) \text{msw}$$

Where:
- $\gamma$ – Performance ratio. Varies between 0 and 1.
- mse – mean square error.
- msw = mean square weight = $\frac{1}{n} \sum_{i=1}^{n} w_i^2$

In this study $\gamma = 0.5$.

**Pseudo Inverse**

Suppose that we use $p$ variables to forecast the peak flood and that the forecast of a given year be calculated using a linear equation, as shown in equation (3):

$$a_{k1}x_1 + a_{k2}x_2 + \cdots + a_{kp}x_p = p_f$$

Where:
- $a_k$ – variable $k$ value
- $x_k$ – variable $k$ coefficient
- $p_f$ – peak flood value of $year_k$

For $Q$ years, we have a linear system with $Q$ equations:

$$a_{11}x_1 + a_{12}x_2 + \cdots + a_{1p}x_p = p_{f1}$$
$$a_{21}x_1 + a_{22}x_2 + \cdots + a_{2p}x_p = p_{f2}$$
$$\vdots$$
$$a_{q1}x_1 + a_{q2}x_2 + \cdots + a_{qp}x_p = p_{fq}$$

The linear system shown in (4) can be represented as:

$$AX = P$$

Where:

$$A = \begin{pmatrix}
a_{11} & a_{12} & \cdots & a_{1p} \\
a_{21} & a_{22} & \cdots & a_{2p} \\
\vdots & \vdots & \ddots & \vdots \\
a_{q1} & a_{q2} & \cdots & a_{qp} \\
\end{pmatrix}$$

If a number of variables, $p$, was equal to $Q$, the linear system shown in equation (5) could be solved using the inverse of $A$, as shown in equation (9):

$$X = A^{-1}P$$

Nevertheless, the number of variables usually used for forecasting $p$ is much lower than $Q$, so the linear system can be solved using equation (9). In can be shown (Albert 72) that a solution

$$X = YP$$

that minimizes the quadratic error given in equation (11) is given by the pseudo-inverse matrix $Y = A^*$.

$$E = \sum_{i=1}^{n}||p_{fi} - A_iX||^2$$

Where:

$$A_i = (a_{i1} & a_{i2} & \cdots & a_{ip})$$

$$A^* = (A^T.A)^{-1}A^T$$

The following algorithm for forecasting with the PI is proposed:

1. Define an input variable set;
2. Select statistically significant $n$ variables;
3. For each year ($year_x$)
   a. Using equation (4) calculate a pseudo-inverse matrix $A^*$ with data of $N-1$ years ($N$ years - $year_x$): $Q = N-1 = 63$.
   b. Calculate a coefficient matrix $X = A^*, P$.
   c. Calculate a forecast value $p_{fk} = A_k . X$.

**Pearson Forecast Correlation Coefficient**

In both forecasting algorithms previously described (FNN and PI), $N$ forecast values, one for each year, are calculated. In this study, using these $N$ forecast values and the corresponding maximum flood level of each year, a Pearson forecast correlation coefficient, $r_f$ is calculated for both FNN and PI methods.

**Results**

**Results using simple correlation and river level in February**

As shown in Table 3, the correlation of the river level in February with the peak flood is $r = 0.761$. To evaluate the power prediction of this variable, a Pearson forecast correlation coefficient, $r_f$ was calculated considering the following procedure: for all the years between 1951 and
2014, 64 linear correlation curves were obtained. For obtaining each curve, the forecasted year is excluded, so only 63 years are considered. Using these correlation curves the peak flood of the forecasted year is then calculated. With the 64 forecasted values the $r_f$ coefficient is calculated. The obtained result is $r_f = 0.745$.

For evaluating both forecasting methods presented in this paper, the following parameters are used: forecasting correlation, $r_f$ and error indices. The error indices are: Ind_1 - number of cases where the difference between the peak flood and the forecasting value is lower than 0.5m; Ind_2 - number of cases where the difference between the peak flood and the forecasting value is situated between 0.5m and 1.0m; Ind_3 - number of cases where the difference between the peak flood and the forecasting value is situated between 1.0m and 1.5m. Ind_4 - number of cases where the difference between the peak flood and the forecasting value is situated above 1.5m.

**Neural Network Results**

To illustrate the method used for forecasting the peak flood of one year with FNN, Figure 2 shows a histogram of the observed errors of 100-peak flood forecasting for the year 1975. One hundred neural networks were trained with random weights and bias initialization. In this example, the following conditions were used for neural networks training: 3 input variables (3 variables with maximum $r$ values in Table 3: river level in February, river level in January and SOI index in January), 6-6-1 architecture and regularization training stop criterion. The training data correspond to the years 1951-2014, except for 1975. As shown, there are errors with absolute values larger than 1. Nevertheless, the mean square error of these 100 predictions is -0.53.

Some experiments were done to determine the best neural network architecture and size of the input variable set. Table 4 shows the obtained results. The neural network training employed two different stopping criteria: mean square error below 0.1 and regularization criterion; input variable sets with 1, 2, 3, 4, 5, 6, 7, and 9 variables orderly by absolute value of Pearson correlation coefficient value, $r$. From Table 3, these orderly list of variables are: river level in February, river level in January, SOI index in January, SOI index in November, SOI index in September, El Nino34 index in January, El Nino34 index in February, El Nino34 index in December and SOI index in December.

**Pseudo Inverse Results**

Table 5 shows the results obtained with a Pseudo Inverse method. Input variable sets with 1, 2, 3, 4, 5, 6, 7, 8 and 9 variables, as previously described.
Discussion and Conclusion

Table 4 shows that the best forecasting coefficient obtained with neural networks was $r_2 = 0.723$. This result was obtained for the regularization training stop method. This result was obtained with only one input variable, the river level in February and with a neural network architecture 4x4x1. The error indices for this table input is: $Ind_1 = 31$, $Ind_2 = 24$, $Ind_3 = 8$ and $Ind_4 = 1$. So, in 51.6% of the years, the peak flood forecasting error was less than 0.5m, and in 91.67% of the years, the peak flood forecasting error was less than 1.0m. The peak flood forecasting was higher than 1.5m in only one year: 1985. For all input set variables, the regularization training stop method shows better results than the mean square error method.

Table 5 shows that the best forecasting coefficient obtained with pseudo inverse method, $r_1 = 0.756$. This result was obtained with four input variables: river level in February, river level in January, SOI index in January and SOI index in November. The error indices for this table input were: $Ind_1 = 37$, $Ind_2 = 22$, $Ind_3 = 4$ and $Ind_4 = 1$. So, in 61.7% of the years, the peak flood forecasting error was less than 0.5m, and in 98.33% of the years, the peak flood forecasting error was less than 1.0m. The peak flood forecasting was higher than 1.5m in only one year: 1985. For all input set variables the Pseudo Inverse method shows better results than neural networks.

In both proposed methods, neural networks and pseudo inverse matrix, there is no methodological bias. In neural networks, the forecasting value is obtained as a mean of 100 different simulations. This is different from other studies that use cross folder validation methods for training and testing neural networks, where the authors, for each folder, try to obtain the neural network with the best performance. In this work, the variable sets were constructed with variables with higher Pearson correlation coefficients. This approach might select variables with redundant information. In future studies, we will address the problem of selecting the best set of variables.

Acknowledgement

Part of the results presented in this paper were obtained through the project for research and human resources qualification, at the undergraduate and graduate level, in the areas of industrial automation, mobile software and Digital TV devices, sponsored by Samsung Eletronica da Amazonia Ltda, under the terms of Brazilian Federal Law number 8.387/91. We like to thank AcademicEnglishSolutions.com for revising the text

REFERENCES


WEB REFERENCES


BIOGRAPHIES

MARCIO MENEZES RODRIGUES received B. Sc. and M.Sc. degrees in Electrical Engineering from the Federal University of Amazonas (UFAM), in 2000 and 2015, respectively. Since 2012, he has been an associate researcher at the Center of Electronic and Information Technology of Federal University of Amazonas. His main research areas are Optimization and Digital Image Processing.

MARLl G uiMARãES FERNANDES COSTA received M.Sc. and D.Sc. degrees in Electrical Engineering from the State University of Campinas (UNICAMP), in 1988 and 2006, respectively. Since 1989, she has been an associate professor at the Electrical Engineering Program of Federal University of Amazonas. Her main research interest areas are Artificial Intelligence and Digital Image Processing.

CICERO FERREIRA FERNANDES COSTA FILHO received M.Sc. and D.Sc. degrees in Electrical Engineering from the State University of Campinas (UNICAMP), in 1986 and 2006, respectively. Since 1989, he has been an associate professor at the Electrical Engineering Program of Federal University of Amazonas. His main research interest areas are Artificial Intelligence and Digital Image Processing.
PERFORMANCE EVALUATION OF DISCONTINUITY TRACKING IN VECTOR FIELDS FOR AVALANCHE SIMULATION

Ángel Rodríguez-Cerro¹, Miguel Lozano², Ignacio García-Fernández² and Rafael J. Martínez-Durá¹
email: angel.rodriguez@uv.es

KEYWORDS
Performance Evaluation, Parallel Computing, Visualisation, Avalanche Simulation

ABSTRACT
This paper presents the acceleration study of a texture based visualisation method able to track discontinuities in a flow, so that it is suitable for interactive environments (eg. games, Virtual Reality simulators). This study analyses the cost of the solution adopted in different scenarios (worst and normal cases) and proposes a new OpenMP version to exploit the current performance of multi-core architectures. A performance evaluation of both, serial and parallel versions, is also carried out in order to show the scalability and speed up achieved. The results show how the conflicts derived from the writing operations limit the speed up and scalability obtained, which are far from the maximum. However, the acceleration obtained by the presented model is useful to take advantage of multi-core architectures in a simple manner and allows to increase the quality associated to the avalanche visualisation.

INTRODUCTION
Simulation and visualisation of fluids are present in video-games, visual effects for feature films and computer animation (Ihmusen et al. (2012), Lever and Komura (2012)). In interactive applications, however, computation time is a key issue and high resolution fluid simulations are not always an option. When interactive frame rates are required, such as in virtual environments, the visualisation of flow often relies on texture based approaches (Laramee et al. (2004)). Texture advection techniques transport textures on a triangle mesh according to the velocity field of a flow. These techniques have been used to visualise a dynamic appearance on the surface of objects and specifically to visualise surface flow in substances such as water, lava, sand or fire (Rasmussen et al. (2004), Bargteil et al. (2006), Kwatra et al. (2007)). Nevertheless, as the texture coordinates are advected, the original texture appearance is lost due to its deformation. For this reason, different strategies have been proposed to control the level of texture deformation (Max and Becker (1996), Neyret (2003), Yu et al. (2009; 2011)).

In these techniques the vector field is supposed to be differentiable (or at least continuous). However, some flow phenomena such as avalanches, some multiphase flows or even dense crowds of people, can show significant discontinuities in the velocity field that lead to undesired texture deformation, as shown in Figure 1. Rodríguez-Cerro et al. (2015) present a technique that enables the visualisation of flows with discontinuities in the spatial domain avoiding the undesired distortion of the texture that appears in such regions. As inputs, that method takes a velocity vector field defined on an arbitrary triangle mesh and a texture, to produce an animation of the flow.

In this paper we analyse the computational cost of the technique proposed by Rodríguez-Cerro et al. (2015), in order to determine its suitability for interactive applications. Upon this analysis, we detect its main bottleneck and propose an acceleration strategy of the method based on OpenMP, to ensure that the method can be used under certain time restrictions, common in real time graphics.

The paper is structured as follows. In this section we briefly describe the flow visualisation technique that we
are going to review and the main tasks involved in the algorithm. In the next Section we evaluate the cost of execution of the algorithm on a serial implementation and determine the bottleneck. Then, we propose an acceleration strategy based on OpenMP, analyse the results and discuss the main limitations found. Finally, we give some concluding remarks and propose future research lines.

Texture advection on flows with discontinuities

We consider a vector field representing the velocity field of a flow, \( \mathbf{v}(\mathbf{x}) \) in a two dimensional regular triangle mesh. For every vertex, \( i \), we shall consider its location, \( \mathbf{x}_i \), and its texture coordinates, \( c_i = (u_i, v_i) \). Let \( i, j \)
be the indexes of two vertexes in the triangle mesh connected by an edge, \( \mathbf{x}_{ij} = \mathbf{x}_j - \mathbf{x}_i \), will be the vector that connects them. In addition, for every vertex we have the value of the velocity field at its location, \( \mathbf{v}_i = \mathbf{v}(\mathbf{x}_i) \). The vector field contains a discontinuity that can be represented as a differentiable curve. We consider that the discontinuity crosses edge \( ij \) if the directional derivative of \( \mathbf{v}(\mathbf{x}) \) along \( \mathbf{x}_{ij} \) is higher than a user defined threshold \( M \). In that case, we add a discontinuity point \( D_{ij} \) to the list of intersections between the discontinuity and the triangle mesh. For every discontinuity point, its location along the edge, \( f_{ij} \in [0, 1] \), and its velocity \( \mathbf{w}_{ij} \) are stored. This velocity does not need to have any relationship with the velocity field defined on the domain, and is estimated as part of the method. The projection of \( \mathbf{w}_{ij} \) on \( \mathbf{x}_{ij} \), and orthogonal to \( \mathbf{w}_{ij} \), will be \( \mathbf{w}_i \) denoted by. Figure 2 shows the variables that are used to describe a discontinuity point.

The algorithm described by Rodríguez-Cerro et al. (2015) tracks the state of the every discontinuity along time during the flow visualization. The process is done in several steps. For the sake of clarity, the steps are not presented here in the exact order they are computed. Later, in this section, the steps are described in the actual order using pseudo-code. In the explanation that follows, labels are defined for different tests and computations involved in the method. These labels will be used later when evaluating the performance.

In a first step, texture coordinates are advected in all the vertexes using the method proposed by Neyret (2003). This step is done for all the vertexes of the triangle mesh and will be labelled as ADVECTION.

Every edge of the triangle mesh is checked to decide if it has a discontinuity. This test is labelled as EXIST. New discontinuity points are created and initialised to a set of default values. Discontinuity points \( D_{ij} \) that already existed in the previous frame are updated by computing

\[
f'_{ij}^{t+\Delta t} = f'_{ij}^{t} + \mathbf{w}_{ij} \Delta t,
\]

where superscript indicates time and \( \Delta t \) is the time between to consecutive frames.

The existing discontinuity points are checked to detect if they have crossed a vertex. If a discontinuity crosses a vertex, say \( j \), then several discontinuity points \( D_{ik,j} \) may converge in that vertex. When this happens, these discontinuity points disappear and new ones appear on edges \( (j, \beta), \beta = k_1, \ldots, k_r \). These new discontinuities have been recently created during the detection step. Figure 3 shows a possible transition of two discontinuity points into a new one across a vertex. In this situation the velocity of the discontinuity \( \mathbf{w}_j \), which is unknown, is estimated solving a least-squares problem of 2 unknowns and \( m \) equations, where \( m \) is the number of converging discontinuity points \( D_{ik,j} \). This step will be labeled as TRANS.DISC.

\[
\text{Figure 2: The variables involved in the estimation of a discontinuity tracked by the algorithm that is parallelized in this work.}
\]

\[
\text{Figure 3: When a discontinuity reaches the end of an edge, the old discontinuity points disappear (b), and a new discontinuity point appear in the edges across the vertex (c).}
\]

In the discontinuity points texture is extrapolated, using an estimation of the texture coordinate gradient. The estimation of the gradient is done for every vertex next
Algorithm 1 Algorithm of the discontinuity tracking and visualisation method

1: for all vertex do
2:   Advect texture coordinates
3:   for all incident edge, e do
4:     if e has discontinuity d then
5:       if d crossed vertex then
6:         Compute least squares from velocities
7:       end if
8:       Estimate velocity and location for d
9:       Extrapolate texture coordinate
10:     end if
11: end for
12: end for
13: Draw

to a discontinuity, by solving a least-squares problem with two unknowns and \( n \) equations, with, at most, \( n \) being equal to the number of edges on the vertex. This step, together with the update of the discontinuity variables, will be labelled as UP_DISC. We can check the main loop of the process in Algorithm 1.

Computational cost analysis

According to the labels defined previously and the pseudo-code presented in Algorithm 1, next we analyse the computational cost of the algorithm depending on the parameters of the problem. In the analysis we shall consider that we have a square scenario, with \( N \times N \) vertices, and that the flow has a discontinuity that affects \( M \) edges.

- ADVECT: Advection of the texture coordinates of every vertex according to the velocity field. In our implementation, we use the method by Neyret (2003). This step is done per vertex, with a cost of \( O(N^2) \). This step is not part of the algorithm object of this analysis. It is independent of whether there are discontinuities or not in the flow.

- EXIST: Detection of the discontinuity between each pair of vertexes directly connected by an edge, performed in line 4 of the algorithm. This operation is done per vertex every frame and requires \( O(N^2) \) floating point operations.

- CHECK_LOST: It detects if a new discontinuity comes from one or more lost discontinuities. This indicates that the lost discontinuities actually have reached and crossed a vertex. This test is performed in line 5 of the algorithm. The cost is \( O(M) \).

- TRANS_DISC: When a discontinuity reaches a vertex and travels from one edge to a different one. This requires the estimation of new velocities and the solution of a least squares problem for every vertex affected. It is performed in line 6 of the algorithm. On a given frame, it can happen that there is no transition, but it can also happen that all edges affected by a discontinuity have a transition. The cost will be, at most, \( O(M) \).

- UP_DISC: For every edge affected by the discontinuity, the velocity and position is updated. Furthermore, the texture advection is extrapolated at both sides of the discontinuity, solving another least squares problem. This is done in lines 8 and 9 of the algorithm and has a cost of \( O(M) \).

According to the previous analysis of the operations that need to be performed, we can conclude that the final cost of the algorithm for every frame will depend on three main factors:

- Number of edges affected by a discontinuity, denoted by \( M \) in the previous discussion. If the discontinuity length is large, we’ll have more points to update their positions.

- Grid size. This is the number of grid points, so when the size of the grid increases then the computation time also increases. As we have considered the domain as a square grid, we measure this factor by the number of vertexes per square side. This leads to a total number of \( N \times N \) vertexes.

- Discontinuity speed. The discontinuity is assumed to travel across the flow domain with certain speed. If the speed is high, then it will cause more transitions of the discontinuity across a vertex, forcing more executions of block TRANS_DISC. This will lead to a higher computational cost in average.

From Algorithm 1 and the previous discussion it can be deduced that some of the steps are not performed every frame. Depending on the state of the discontinuity, in some situations the estimation of velocities performed in task labelled as TRANS_DISC are not executed; in the worst case, could be executed for every edge affected by the discontinuity but it can happen that there is no vertex transition in a given frame. In order to evaluate empirically the cost of the method we have designed two scenarios; a worst case scenario and an standard scenario.

The worst case scenario is one for which the highest possible number of edges have a discontinuity and for which every frame TRANS_DISC is executed at every discontinuity. To achieve this, we have used a velocity field that has four different values, forming a checkerboard in the mesh grid, as shown in Figure 4. The velocity field changes after every frame, shifting all the velocities one vertex in both \( x \) and \( y \) direction. This causes that \( M \) grows with \( N \times N \) and that every frame TRANS_DISC needs to be executed for every edge with discontinuity.
The standard or average scenario corresponds to a discontinuity crossing the domain of the flow. In the work by Rodríguez-Cerro et al. (2015), discontinuities are assumed to be differentiable curves, and they are approximated as straight segments, tangent to the curve. In order to simplify the scenario, and have an stable number of discontinuities, we have decided to use a straight discontinuity. We shall call this scenario a normal avalanche case as its behaviour is similar to the situation that arises during the visualisation of an avalanche.

PERFORMANCE EVALUATION

The visualisation method described in the previous section takes as input a vector field and performs two main tasks. First, texture coordinates are advected according to the vector field and, second, discontinuities are detected, tracked and visualised. In the previous section, we have labelled the first task as ADVECT. The second task comprises the subtasks EXIST, TRANS.DISC, UP.DISC and CHECK.LOST. This second task will be referred to as StepDiscontinuity. The time consumed by these tasks and also the subtasks involved in updating the discontinuities for both scenarios (normal and worst cases) are shown in Figures 5 and 6. These times correspond to the serial version of the C++ implementation carried out to compute the model. In order to average the data presented we use 800 simulation cycles executed on a multi-core computer with 4 Intel Xeon E5-2650 @2.00Ghz CPU and 32GB RAM.

Both figures show that discontinuity updates (StepDiscontinuity) are clearly more expensive than advecting textures along the flow defined by the input vector field. It is noteworthy to recall that this visualisation method is addressed to interactive scenarios, which must run at fixed frequency of 60Hz. For this reason, each step of the whole process must be computed in less than 1/60 seconds (0.017 secs approx.). This imposes a hard temporal constraint to deal with, and is a key motivation to address the acceleration of the algorithm described in Rodríguez-Cerro et al. (2015).

Considering the aforementioned temporal constraint, the average step times obtained for the normal case indicate that the mesh size cannot be scaled beyond 80x80 nodes. In this situation, the task identified as the most expensive has been TRANS.DISC. We have also measured the average step times for the worst case scenario, and the results show how this situation is harder to compute than the normal case; in this case we can only manage meshes of 40x40 nodes according to the time requirements. This second test also confirms that the bottleneck is associated to task TRANS.DISC. If we compare these results with the theoretical analysis developed in the previous section, we can see that there are other tasks with higher theoretical cost associated than TRANS.DISC. However, its cost per discontinuity is high compared to the cost per vertex of EXIST or UP.DISC. Thus, for the size meshes we are considering, the results show how the real cost of TRANS.DISC is clearly higher than any other subtask involved.

Once the profiling of the main tasks has been carried out, the next section aims to reduce the computational time required by TRANS.DISC task, as it has been
identified as the current bottleneck of the visualisation method.

PARALLEL DESIGN AND RESULTS

In order to accelerate the current bottleneck of our visualisation system we have introduced a set of OpenMP (Dagum and Enon (1998), ope (2015)) threads to work concurrently in this task. The OpenMP design proposed introduces a new parallel for-section to work with the mesh points (vertices). As in other eulerian (grid based) approaches like the method analysed in this work, parallel design is based on a per-vertex organisation. Our implementation uses a multi-threading environment programmed with OpenMP so all the threads share the data structures of the problem which are allocated in the main memory. All the tests have been executed on a multi-core computer with 4 Intel Xeon E5-2650 @2.00GHz CPU, with 4 cores each, and the possibility of 2 threads per core, using HyperThreadingTM. It has 32GB RAM and it has been also used in the previous section.

Since the load associated to the grid nodes is not homogeneous (it mainly depends on the number of discontinuities associated to a vertex and whether the discontinuity is crossing a vertex), to evaluate the degree of balance associated to our problem we have also evaluated the scheduling options that openMP provides to organise the workload within a multi-threading environment. In our current design, one thread is going to work or update a mesh node, which implies to perform some writing operations over the main shared structure (mesh). When a set of threads tries to write concurrently in a shared structure the data dependencies (conflicts) will force the system to serialise these operations. Thus, according to the serial design used and the OpenMP strategy, a linear speed up is going to be impossible to obtain. The Figure 7 shows the speed up obtained in the normal case for the TRANS.DISC sub-task with a 160x160 size mesh.

Figure 7: Speed up obtained for the TRANS.DISC task in the normal avalanche scenario.

Considering the shared memory multiprocessor architecture we are using in this work (4 CPU, 4 cores/CPU, 2 threads/core), we could expect a maximum (linear) speed up of 16x. However, this speed up cannot be achieved, as explained before, due to serialisation of some writing operations. Figure 7 shows that the maximum speed up obtained by static and guided scheduling strategies is 4, while dynamic scheduling can only accelerate 2.5 times the serial version. In the normal case, there are only a few mesh-nodes (threads) that have workload to compute and this load is not high, as TRANS.DISC is not executed every frame. Furthermore, these threads are trying to complete their tasks in a parallel way, however the high number of conflicts derived from the writing operations serialise part of their work. As a consequence of this, the speed up obtained for the normal case is reduced and the figure also shows how this situation is still worst for dynamic strategies, as they must also pay an extra time for scheduling every time a thread complete its tasks.

Figure 8 shows the speed up obtained for the worst case simulated. In this case a dynamic scheduling strategy is nearly 8 times faster than the serial version (16 is the maximum expected), while static approaches are around 6 in this scenario. Considering the previous case, now the situation has changed mainly due to the current workload now is high and unbalanced (not all the nodes have the same work), so dynamic scheduling can exploit this to finally achieve a better speed up.

Figure 8: Speed up obtained for task TRANS.DISC in the worst case scenario.

To analyse the scalability associated to this approach we have also calculated the efficiency obtained when accelerating task TRANS.DISC for the worst case. Figure 9 shows how the system can properly scale up to 10 cores, as efficiency is higher than 0.5 within this range. It is not worth it, however, to use a higher number of cores, since they are not going to provide further acceleration.

Figure 10 contains a time comparison between the parallel version (worst case, 16 threads and 60x60 mesh size) and the serial one in order to show the global acceleration obtained under different domains, but keeping the same final results. Here, the Total-time indicates the required time to solve the whole problem, while Advect and StepDiscontinuity indicate the time consumed by the two main sub-tasks considered. As the figure shows, the global acceleration achieved in the parallel version mainly depends on the performance obtained while updating the discontinuities (StepDisc).
CONCLUSIONS AND FUTURE WORK

This work introduces a parallel approach based on OpenMP, to accelerate a texture based avalanche visualisation models that handles discontinuities in the velocity field. The application context of the method (interactive computer graphics applications) introduces a hard temporal constraint associated to the average cycle time (60 Hz) that must be met.

The experiments carried out for two scenarios (an average case and a worst case scenario) show how that threads cannot fully exploit the current performance of multi-core architectures due to the conflicts derived from the writing operations. As these conflicts must be handled, the operations finally are serialised so the degree of parallelism obtained is reduced within the corresponding speed up. To avoid this conflicts we must undertake major changes in the serial code. The first idea is to associate blocks of nodes (submatrices of nodes from the mesh) to threads workload instead of single ones. This will reduce the writing conflicts only to the border nodes, that can be managed independently.

The present work shows a very useful and simple manner to obtain the necessary acceleration to be able to nearly duplicate the size of the meshes involved. This aspect will have a clear impact on the quality associated to the avalanche visualisation using the studied method.

ACKNOWLEDGEMENT

This research has been partially supported by the Spanish MINECO Grant RACHEL TEC2013-47141-C4-4-R.

REFERENCES


WATER DISTRIBUTION SYSTEMS
MODEL CALIBRATION AND HYDRAULIC OPTIMIZATION OF THE WATER DISTRIBUTION SYSTEM OF THE UPPER SILESIAN WATERWORKS

Jan Studzinski, Andrzej Ziółkowski, Dorota Tomasiuk
Polish Academy of Sciences, Systems Research Institute
Nowelska 6, 01-447 Warsaw, Poland
studzins@ibspan.waw.pl

KEYWORDS:
Drink water distribution system, water net hydraulic model, hydraulic optimization, water net management.

ABSTRACT
For several years some integrated ICT system for complex management of water and wastewater communal networks are under development at the Systems Research Institute (IBS PAN) in Warsaw. Until now they were applied in small and medium waterworks in Poland but the last application concerns the biggest Polish waterworks – the Upper Silesian Water Distribution System (GPW). In the paper ICT system MOSKAN-W assigned for the waterworks and two optimization algorithms implemented in it and designed for automatic calibration of the water net hydraulic model and for network hydraulic optimization are presented.

INTRODUCTION
The management of municipal waterworks consisting of water take out and water clarification stations, of water and wastewater networks and of a wastewater treatment plant is a very complicated task whose realization needs skilled operators and several computer programs for solving differentiated technical and economical questions. The present practice of waterworks management relies in regarding all enterprise objects as separated items which are operated independently by experienced workers doing many of their actions by hand. Such approach needs their permanent presence in-house and it is unfortunately very time consuming and inaccurate. A better solution is the use of complex ICT systems that can manage all waterworks objects as one integrated system and do it automatically what means faster and exacter. Such systems are under development for several years at IBS PAN in Warsaw and they are dedicated for municipal water and wastewater networks. Their standard structure consists of four or five modules depending on their destination, i.e. of a GIS (Geographical Information System) system, SCADA (System of Control and Diagnostics Analysis) system, CIS (Customer Information System) or AMR (Automatic Meter Reading) systems in case of water nets, of a network hydraulic model and of a computational module in which some modeling, optimization and approximation algorithms are collected (Stachura et al, 2012, Studzinski, 2012). The ICT systems have been used so far in several Polish communal waterworks of small and medium scale. The algorithms in their computational modules solved the tasks of hydraulic model calibration and of hydraulic optimization and of net revitalization and planning in case of wastewater networks (Sluzalec et al, 2013) and in case of water networks they solved the tasks of model calibration, net optimization and revitalization, of pumps control and of hidden leaks detection (Fajdek et al, 2015, Rojek and Studzinski, 2014, Stachura et al, 20145, Studzinski, 2014). The networks in these waterworks are rather small and the hydraulic and optimization calculation of them are running fast, smoothly and easy. The calculation problems arise when the networkcs are large as in the case of the Upper Silesian water distribution net what is the topic of the research project realized now at IBS PAN. In the following these problems, the ideas to solve them and the object investigated are described.

ICT SYSTEM FOR UPPER SILESIAN WATERWORKS MANAGEMENT
The investigated water net is the Upper Silesian water supply and distribution system that is the biggest waterworks in Poland and one of the biggest ones in Europe. Its activity area is the whole Upper Silesian region with the surface of 4.300.000 km² (Fig. 1). At the water net are several network objects installed which are: 19 pump stations – among them 11 pump works for water take out and 8 pump works for water pressure raising which are localized inside the network, 11 water clarification stations and 9 storage reservoirs with the entire water volume of ca. 400.000 m³. The other features of the waterworks are following:
• the water net has mainly the ring structure and its whole length is 900 km;
• the pipe diameters are untypical for municipal water nets and their sizes are between 300 and 1.800 mm;
• the network end users are in 90% not individual but bulk purchasers, i.e. the Upper Silesian cities;
• the water transported over the network is won partly from the surface reservoirs (7 works) and partly from the wells (4 works);
• the daily water production of the waterworks is 400.000 m³;
• the Upper Silesian region is characterized by the large differences in altitude which are up to 120 m;
• at the water net is a telemetric SCADA system installed consisting of 150 measurement points with the devises to measure the water flows and pressures;
• in particular network points are also the measuring tools installed to measure the quality of the drink water, i.e. its turbidity and the distribution of the chlorine concentration in the net;
• altogether ca. 500 sensors are installed at the water net;
at the network is also an AMR system mounted for on-line measurements and registration of the water amounts which are sold to the network users; almost all water purchasers in the number of 10,000 are already connected to the system.

**Figure 1.** The water distribution system supplying with drink water the Upper Silesian region.

The ICT system developed for GPW is shown in Fig. 2. It consists presently of GIS and two monitoring systems (SCADA and AMR or CIS which was in use before AMR came), of the network hydraulic model and of two optimization algorithms for model calibration and for network hydraulic optimization (Studzinski, 2015). The ICT system is still under development and including in it other algorithms for solving the water net management tasks like network revitalization or pump stations control is expected in the next future (Wojtowicz et al., 2014B). The differences between this ICT system and the other ones which have been prepared for small and medium waterworks are caused by the large scale of the GPW net and they are as follows: in GIS system some algorithms are included to simplify the network graphs which are exported to the hydraulic model; the hydraulic model is made up in form of an Internet application and not as desktop one; in the optimization programs the old heuristic algorithm has been replaced by a genetic one. The first and third changes are made in order to make possible and to accelerate the calculations of optimization which are practically unworkable with the network graph consisting initially of 4,000 pipes and 4,000 nodes and using the algorithm that needs ca. 10,000 runs to get the optimum by only 1 objective function. The full graph of the water net is exported from GIS to the hydraulic model when only single simulation of the network will be made. The second change is made to facilitate the job of the network workers who must operate while doing their actions on the large space of 4,300,000 km². In the new program called MOSKAN-W for hydraulic calculations of the network the EPANET source codes are used (Sluzalec et al, 2014). The cooperation between GIS and MOSKAN-W is shown in Fig.3.

**Figure 2.** Block diagram of the ICT system for the water net management.

**Figure 3.** integration between GIS system and the water net hydraulic model.

**ALGORITHMS OF OPTIMIZATION**

**Hydraulic model calibration**

Calibration procedure in case of water nets consists usually in changing the roughness values of the network pipes in such the way that flows and pressures measured and calculated are possibly the same in the net points where the sensors of SCADA system have been installed. This changing occurs normally by hand for in the waterworks there are not the programs that could support this action by automatic computing (Wojtowicz et al, 2014A).

**Figure 4.** Example of preparation of data for the calibration algorithm.
The algorithm presented below executes the calibration procedure in three following steps:
1. Preparation of the initial data consisting in division all network pipes in groups depending on pipe diameters, age and material.
2. Changing the roughness of pipes regarding the pipe groups and not individual pipes.
3. If the roughness change in a group exceeds the values field given then changing there the nominal pipe diameters; this change occurs either in frame of a given values field.

In this way the algorithm has got two phases of calculation regarding the roughness and diameter changes that follow one after another.

In Figures 5 and 6 the exemplary water net model and the data preparation for its calibration are shown. The net consists of 25 pipes of the same age and made of the same material and on two pipes (4 and 5) and two nodes (7 and 2) of the net the measuring devices for flow and pressure are installed. In Fig. 7 one can see the diagrams of calculated and measured flow and pressure values designed for 24 hours and shown for one pipe and one node before the calibration run; the initial roughness of all pipes is 0.3 mm. The pipes are divided into 2 groups regarding their diameters (18 pipes with the diameters of 100 and 150 mm are included into 1st group and the rest of the pipes with the diameters of 200, 250 and 300 mm are included into 2nd group). By the calibration only the roughness values in two pipe groups will be changed and the acceptable fields for changing the values is between 0.2 and 1.8 for group 1 and between 0.3 and 2.0 for group 2.

The results of the calibration are shown in Figures 8 and 9. The genetic algorithm made 39 optimization runs and the roughness of the pipes in two groups has reached the value 1.5 mm (to see in yellow strips in Fig. 8). One can see in Fig. 9 that the pressure values in node 2 and flow values in pipe 5 are the same for the calculation results and the measurement data what allows to consider the calibration algorithm to be effective and also very fast by the reason of using the genetic algorithm of optimization.

**Water net hydraulic optimization**

The second algorithm supporting the management of GPW concerns the hydraulic optimization of the water network by
means of exchange of particular network pipes and/or of control of pumps in the water take out stations or in the works raising the water pressure within the water net. In the algorithms solving the task in small and medium waterworks the calculation was done for all pump stations in the same run for there are not more than only several pumps in such the enterprises.

In GPW the situation is more complicated because there are 19 pump stations and 120 pumps in them and finding the control schemes for all devices simultaneously is practically not possible. Because of that the algorithm proposed consists of two stages when on the 1st stage the controls are calculated for the pumps stations seen as single generalized pumps and the 2nd stage the calculation is done for each pumps station and for the pumps there individually. It means that in the case of GPW on the 1st stage of the algorithm the 24-hours diagrams of pressures and flows for 19 generalized pumps will be calculated and on the 2nd stage 19 separated problems will be solved, i.e. the individual controls for the pumps working in each station are to find. Such the division of the hydraulic optimization task in two separated stages makes the problem solvable from the computational point of view. In the following the realization of the 2nd stage of the algorithm done for a real pumps station of GPW will be described.

The object calculated is 1 of 19 pump stations of GPW in which only 1 pump no. 1 works and from which 2 pipes numbered p1 and p23 go out; they are ended with 2 nodes numbered n2 and n29. The pressure values in these nodes are too small against the values that have been calculated on the 1st stage of the algorithm. The problem is to find out the pipes with new diameters and to calculate the pump velocity in such the way that the obtained node pressures will fit to the indicated earlier value areas.

Figure 10. View on the pumps station calculated and the parameters of the pump to be controlled.

In Figures 10 and 11 the scheme of the pipe connections in the pumps station investigation as well as the characteristics of the pump and of the nodes concerned are shown. In Figures 12 and 13 the screens of the MOSKAN-W program prepared for introducing the input data for changing the pump velocity and the pipe diameters are to see. In this example the pump velocity can be change in the area between 60% and 100% of its nominal speed, the new pipes can have their diameters between 100 mm and 1,200 mm and the acceptable or preferable node pressures are lying between 20 m and 70 m or 28 m and 36 m for node n2 and between 10 m and 70 m or 37 m and 45 m for node n25, respectively. To solve the problem a genetic algorithm of optimization with the use of fuzzy sets while calculating the node pressures is applied; the fuzzy sets are used to make a distinction between the acceptable and preferable value areas while calculating the node pressures.

Figure 11. Parameters of two nodes being the outputs of the pump station calculated.

Figure 12. Preparation of input data for the pump control.
Figure 13. Preparation of input data for the exchange of pipes.

In Figures 14 and 15 the results of the hydraulic optimization made for the single pumps station are given. The pressure values at the end nodes n2 and n29 of the pumps work raised from 11.82 m to 35.86 m and to 37.24 m, respectively, what is the consequence of the pipe diameters change from 600 mm to 488 mm for p1 and from 800 mm to 1.048 mm for p23, and of the pump velocity change from 0.76 % to 0.89 % of its nominal speed. The change of the pipe diameters and of pump velocity caused not only the change of the node pressures but also the alteration of the water flow in the pipes p1 and p23 that is now smaller in p1 and bigger in p23; in the first pipe the flow speed is reduced from 1.18 m/sec to 0.80 m/sec and in the second one it is raised from 1.72 m/sec to 2.52 m/sec.

Figure 14. Results of the hydraulic optimization of the pumps station calculated.

Figure 15. Comparison of the data concerning the pumps station calculated before and after the optimization.

CONCLUSIONS

In the paper the ICT system developed for the management of the water distribution network of Upper Silesian waterworks GPW is presented and two optimization algorithms implemented in it and designed for automatic calibration of the water net hydraulic model and for hydraulic optimization of the network are described.

The idea to develop a new ICT system and new computational algorithms for GPW has been caused by its large number of pipes and nodes and by the big number of the network objects what resulted with problems by the hydraulic and optimization calculation of the water net. To avoid the problems some mechanisms to simplify the network graph exported to the hydraulic model have been implemented into GIS system and also several changes have been made in the algorithms to accelerate and to make exact their calculation. These changes consist in using a genetic algorithm for optimization instead of heuristic one, in dividing the algorithm of hydraulic optimization into two stages and in applying the fuzzy sets for better distinguishing between the acceptable and preferable values by looking for the control parameters. The examples of computation presented show efficiency of the algorithms improved and correctness of the changes made.

Acknowledgement

The paper presented has been realized in the frame of the research project co-financed by the European Union from the European Regional Development Fund, Sub-measure 1.3.1. "Development Projects"; project title: "IT system supporting the optimization and planning of production and distribution of water intended for human consumption in the sub-region of the central and western province of Silesia"; project ref no POIG.01.03.01-14-034/12.
REFERENCES


Wójciszowicz P., Pawlak A., Studzinski J.: Preliminary results of hydraulic modelling and calibration of the Upper Silesian Waterworks in Poland. 11th International Conference on Hydroinformatics HIC 2014, NY City, USA, 2014A.


BIOGRAPHIES

Jan STUDZINSKI, born in 1946 in Warsaw, Poland, educated at the Technical University of Warsaw at the Faculty of Electrical Engineering and at the Warsaw University at the Faculty of Mathematics. He is working at the Systems Research Institute of Polish Academy of Sciences, where he has obtained his PhD. and Sc.D. degrees and where he leads the Center for Applications of Informatics in Environmental Engineering. Author or Coauthor of 4 books and more than 250 scientific papers, he is dealing for many years with mathematical modeling and computer simulation of complex dynamical systems, with development of optimization and control methods and of computer aided decision support systems for management of communal waterworks. Board chairman of the Foundation for Development of System Sciences of Polish Academy of Sciences.

Dorota TOMASIUK, born in 1982 in Warsaw, educated at WIT University – Warsaw Informatics Technology. She is working at the Systems Research Institute of Polish Academy of Sciences, where she is dealing with applications of the computer aided management systems in waterworks and with the management of research projects.

Andrzej ZIÓŁKOWSKI, born in 1948 in Warsaw, educated at the Technical University of Warsaw at the Faculty of Electronic Engineering. He was working for some years at the Systems Research Institute of Polish Academy of Sciences, where he has obtained his PhD. degree and afterwards he grounded his own microenterprise AZ Soft dealing with programming of complex ICT systems for waterworks management.
A SIMULATION STUDY OF MINIMIZING WATER COST IN MULTISOURCE WATER DISTRIBUTION SYSTEM

Rafal Brodziak  
Institute of Environmental Engineering  
Poznań University of Technology  
ul. Piotrowo 3a, 60-965 Poznań, Poland  
E-mail: Rafal.Brodziak@put.poznan.pl

Andrzej Urbaniaik  
Institute of Computing Science  
Poznań University of Technology  
ul. Piotrowo 2, 60-965 Poznań, Poland  
E-mail: Andrzej.Urbaniaik@cs.put.poznan.pl

Jędrzej Byłka  
Institute of Environmental Engineering  
Poznań University of Technology  
ul. Piotrowo 3a, 60-965 Poznań, Poland  
E-mail: jedrzejbylka@gmail.com

Przemysław Zakrzewski  
Institute of Computing Science  
Poznań University of Technology  
ul. Piotrowo 2, 60-965 Poznań, Poland  
E-mail: Przemyslaw.Zakrzewski@cs.put.poznan.pl

KEYWORDS
modelling of water supply system, control of multisource water distribution system,

ABSTRACT
The paper presents an example of using simulation to optimize the cost of water supply system in case when its uses several sources of water. The article presents the characteristics of water supply systems with multiple sources and example of control criteria of such networks. Optimization of water supply networks requires building the model and taking into account basic physical laws. A sample water supply system consisting of several sources of water was analyzed. Calculations were performed in accordance to proposed steps in the environment using the EPANET and MATLAB software. The method proposed to seek solutions allowed to obtain the minimum cost for the analyzed cases.

WATER SUPPLY SYSTEM WITH MULTIPLE SOURCES
The purpose of the water supply system is to meet the demand for water quality at a suitable pressure, in a continuous and reliable manner. Due to limitations related to water resources and to the water distribution network possibilities, water supply systems often are supplied from more than one source. In order to increase system reliability, these sources should use the independent resources of groundwater or surface water. One of the main problems associated with the use of multiple sources of operating water supply system is control - determining how much water will be pumped from each of the sources. All sources might differ in terms of water resources, water quality and the cost of pumping and purification. The criteria used for determining control algorithms can be either:

- economic - to increase the water flow pumped into the network from the cheapest sources and reduce the water flow from sources which are most expensive,
- quality - to increase the water flow pumped into the network from sources with the best water quality and to reduce the water flow from lower quality water sources.

In the control process in addition to the optimization criteria can be identified constraints that must be fulfilled. Management criteria in water supply system can have technical, economical, environmental or legal nature (Brodziak and Byłka 2014). The main limitation of the water supply control system is the need to ensure water with adequate quantity and at the correct pressure for all customers. Others restrictions may arise for the following reasons eg.:

- due to technical capabilities of equipment (maximum capacity of pumps, regulators, filters, etc.), control capability (eg. inverters, valves) and the production capacity of the water treatment plant (the maximum possible and the minimum necessary water quantity to ensure the continuity of the production process at the water treatment plant),
- related to the legal water permit granted for a given source - which determine the maximum water flow from groundwater resources or surface reservoirs,
- because of the obligations arising from the agreements - if a company buys water from external sources (from other companies) in the contract can be established minimum and maximum amount of water to be taken from a particular source.

Due to the complex nature of process and variety of the criteria, there is a need to develop control methods that will allow to use sources of water supply in an efficient way. Operation scheduling of water supply system can be supported through the use of hydraulic models water distribution networks and methods of optimization.

WATER DISTRIBUTION NETWORK MODELING
Mathematical modeling allows for mapping operations states of the water network at different boundary conditions. Through the models of water supply network different operation scenarios can be check and compare each other. Elements of water supply network which can be mapped in the model are: pipelines (for calculating flows and pressure losses), nodes (for assuming water extraction in time steps), tanks and reservoirs (for assuming water level as boundary...
conditions), pumps (for assuming the lifting height characteristics in the flow function) and valves. Fundamental rights, to be used for the construction mathematical models of water supply networks are (Boulos et. al. 2006):

- The principle of conservation of mass, whereby the amount of water that flows in to the node ($\Sigma Q_{in}$) equals the amount of discharge water ($\Sigma Q_{out}$) minus the amount of water which is extracted in it ($q_i$). The principle of conservation of mass can be described by the equation:

$$\Sigma Q_{in} - \Sigma Q_{out} = q_i \quad (1)$$

- The principles of conservation of energy, whereby between two nodes A and B connected to each other by a pipeline, the pressure loss can be written as follows:

$$H_A - H_B = h_{A,B} = \Sigma K_i [Q_i]^n \quad (2)$$

Where $H_A$ and $H_B$ is the total energy at node A and B and $h_{A,B}$ is the total loss of the section between nodes A and B. The sum of the losses can be expressed by a coefficient loss on the section which depends on the flow raised to the power $n$.

Principles of mass and energy conservation are the basis for building a mathematical model of water supply network. The model should include parameters of water transport systems, water partitions in nodes and pressures on system boundaries. The mathematical model of water distribution network is a non-linear set of equations, in the solution we obtain data: flow in pipes and pressure in nodes. The system consists of $N_e-1$ equations for the $N_e$ nodes and $N_l$ equations for the $N_l$ loops. Equations are solved with numerical methods - usually using Newton-Raphson or gradient method. In practice, model calculation of pipe network are performed using computer software with graphical user interface. One of the most known software used for modeling water networks is Epanet, program developed by the US EPA (Rossman 2000). This software is available as the public domain, along with the code sources and Dynamic-Link Library.

**EXAMPLE OF NETWORK AND CONTROL PRINCIPLES**

For analysis purpose an exemplary model of water supply network was built (Figure 1). The network consists of: 2 loops, 14 pipelines, 8 junction, 3 sources of water supply (A, B, C) and 2 valves that allow to control the volume of water supply from sources B and C. The model assumes a constant demand for water in junctions, variability of water consumption during the day wasn’t implemented. Source A does not have a valve, as is the source held by the water supply company, it have good water quality and cheapest extraction cost. The source B and source C are external sources, from which the company buys water. In given example water sources are modeled as reservoirs with constant water level. The network operator may regulate the flow of water from external sources by increasing or decreasing the resistance of valves. External sources also have good quality water, but higher cost of extractions. From the economical point of view of the network operator, the best solution would be to used only water from its own source, but it is not possible for technical reasons. Source A is not sufficient to meet the demand for water for all consumers. Therefore, the operator need to use external water supply sources.

![Figure 1. Exemplary water supply network with 2 rings, 14 pipelines, 8 nodes and 3 sources.](image)

The purpose of operator is controlling and determination of valves settings $Z_i$ (value of throttling - mH$_2$O) to extract as much water from the cheapest sources and the least with the most expensive sources, while meeting constraints. The objective function can be formulated as follows:

$$\min(K_z = \Sigma K_i Q_i) \quad (3)$$

where:

- $K_z$ – total cost [PLN],
- $K_i$ - cost per unit of water from the i-th source [PLN/m3],
- $Q_i$ - the quantity of water extracted from the i-th source [m3]

The constraint is the need to ensure the supply of water in each point in the network at a required pressure:

$$H_i > H_w \quad (4)$$

where:

- $H_i$ - pressure in each node [mH$_2$O],
- $H_w$ - minimal required water pressure [mH$_2$O]

Computer environment calculating the pressure on each node assumes various conditions of the water supply network - for different scenarios control of valves. The simulation aim was to find the most economically effective control scenario with fulfill the conditions described by equation 3 and 4. Calculations were carried out in the following steps:
1. Describe the network structure, the partitions in the nodes and pressure in the water supply nodes - constant model data.
2. Establish the initial conditions set of valves $Z_i$ – set up fully closed valves of the source of B and C, the water was pumped only from source A.
3. Calculate the pressure in nodes (H), and the total cost of water $K_z$ (equations 1-3)  
4. Change the settings for the valves sources B and C in steps of 0.05 m H2O loss and repeat calculations with step 3. 
4. Repeat step 4 until fully open (0 mH2O throttling) valves B and C.  
5. Reject obtained solutions which not fulfilled constraint (equation 4)  
6. Among obtained acceptable solutions find the optimal set of valves ($Z_i$) set with minimal total cost.

RESULTS OF CASE STUDY

The calculations were implemented using the following tools: 
- Epanet 
- Matlab 
- Epanet-Matlab toolkit.

Epanet solves the equations of mathematical model. Epanet - Matlab toolkit (web1), allows to execute functions of Epanet libraries within Matlab and load file describing structure of the network (step 1). Script prepared in Matlab control the setting of valves changes (step 2 and 4), calculate the value of the objective function for individual cases and save the results of the pressure nodes for each combination (step 3). The result of the calculation is a matrix in which are stored the pressure at each node and total price of water for each combination. Solutions matrix is searched, results which do not meet the criteria are rejected (step 5) and results with the lowest total cost can be found(step 6). The calculations were performed for three exemplary cases for which the cost of water in each source are listed in Table 1. For each case, the water for source A was assumed as fixed price, assuming that it is source owned by the water company. The differences between the cases arising from the various water costs purchased from external suppliers (source B and source C).

Table 1. The water unit cost for each source

<table>
<thead>
<tr>
<th></th>
<th>$K_A$ [PLN/m3]</th>
<th>$K_B$ [PLN/m3]</th>
<th>$K_C$ [PLN/m3]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE 1</td>
<td>0,4</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>CASE 2</td>
<td>0,4</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>CASE 3</td>
<td>0,7</td>
<td>0,7</td>
<td>0,7</td>
</tr>
</tbody>
</table>

Table 2. Total cost of water and valve setting in analysed cases.

<table>
<thead>
<tr>
<th></th>
<th>$K_z$ [PLN]</th>
<th>$Z_A$ [mH2O]</th>
<th>$Z_B$ [mH2O]</th>
<th>$Z_C$ [mH2O]</th>
</tr>
</thead>
<tbody>
<tr>
<td>CASE 1</td>
<td>186,89</td>
<td>0</td>
<td>19,85</td>
<td>16,75</td>
</tr>
<tr>
<td>CASE 2</td>
<td>188,25</td>
<td>0</td>
<td>20,00</td>
<td>15,10</td>
</tr>
<tr>
<td>CASE 3</td>
<td>72,52</td>
<td>0</td>
<td>19,85</td>
<td>16,75</td>
</tr>
</tbody>
</table>

Minimum total cost and the valve settings for each cases as optimal results are summarized in Table 2. The results obtained in subsequent stages of simulations for variable valve setting values can be represented in the graphs as a function of $K_z$. In this way can be presented feasible solutions for objective function and constraint. Charts in

Figure 2 presented results for CASE 1 – the total cost $K_z$ as a function of settings valve B($Z_B$) and ($Z_C$).

Figure 2. Charts of total cost $K_z$ as a function of valve B and C settings for CASE 1

CONCLUSION

1. Water supply systems are often supplied from multiple sources in order to improve network conditions and reduce the cost of water to consumers. Strategy for extracting water from different sources should be developed, to determine which sources will be used to achieve maximum efficiency. Practical implementation of this strategy required to find appropriate control scenarios. Scenarios can be simulated and compared using computer models (Urbaniak et. al. 2015). In the paper, we present a tool that allows to analyze all the possible scenarios of control and through it to find the optimum solution.

2. Calculations were performed for all possible combinations of valve settings. This approach requires a multiple computation, but it gives guarantees to finding the best possible solution. For analyzed simple water supply model assume 0.05 m H2O loss step for changing the settings of sources B and C. These assumptions allowed to achieve detailed grid of local solutions with the simulation time of 15 minutes on PC with Intel Core i5-2500 CPU @ 3.3 GHz and 16 GB RAM.
3. Simulations for the real water network, which can consist of several thousands of nodes and hundreds of loops, through a method of solving all combinations can be time-consuming task. There are a several optimization methods for water distribution system controls. The most commonly used are meta-heuristic methods (Yang et al. 2012), especially genetic algorithms (Nicklow et al. 2010). For this reason, the authors intend to use this kinds of methods to find the optimal solution in future works.

4. Presented method is based on a complete review of all possible solutions. Prepared tool will be able to serve as a network benchmark for comparing the results of different optimization methods in controlling multisource water supply network.

REFERENCES


WEB REFERENCES
web1 - EPANET MATLAB Toolkit - File Exchange - MATLAB Central, Available at:

AUTHOR BIOGRAPHIES
RAFAŁ BRODZIAK received his Engineering degree from the Poznan University of Technology (Faculty of Computing Science and Management) at field of Automation in year 2008 and Master degree at Microcomputer Systems of Control in 2009. In year 2010 he took internship at the University of Tokyo in the Department of Electrical Engineering and Information Systems, where preparing proposal of tutorial program to learn motion control for undergraduate students. Since graduating he is working at Institute of Environmental Engineering as teaching assistant and giving classes in automation and computer-aided design systems. In last years he has been involved in projects in Division of Water Supply and Environment Protection at Poznan University of Technology where currently working on his PhD thesis. He is developing his research field in the control of water distributions system, water intake and water treatment processes.

JEDRZEJ BYLKA was born in Poznan, Poland, in year 2012 graduated from Poznan University of Technology (MSC of Environmental engineering) and Adam Mickiewicz University (BE in philosophy). After the scholarship on Polish Academy of Sciences (System Research Institute), he is working in Institute of Environmental Engineering (Poznan University of Technology).

PRZEMYSŁAW ZAKRZEWSKI received his Ph.D. from the Poznan University of Technology at Faculty of Computing Science and Management at field of Automation and Robotics in year 2005. The title of his dissertation is "Algorithms for integrated control of the wastewater treatment process using activated sludge". Since 1996 he is working at Institute of Computing Science at the Poznan University of Technology. His main research area is the control, modelling and optimization of wastewater treatment processes. Recently he is involved in design and implementation of control systems using the Industrial Internet of Things architecture.

Andrzej URBANIAK was born in Poland and graduated at Poznan University of Technology (Control Engineering) and Poznan University of A. Mickiewicz (Mathematics). He obtained the PhD degree in 1979. From 1990 he is a professor of Institute of Computing Science. He is author or co-author of 5 books and over 200 papers concerning the computer control systems and application of computer science in environmental engineering.
MODELLING AND OPTIMIZATION OF COMMUNAL SEWERAGE SYSTEMS

Grazyna Petriczek, Jan Studzinski
Systems Research Institute of the Polish Academy of Sciences
Newelska 6, 01-447 Warsaw, Poland
studzins@ibspan.waw.pl, makur@xl.wp.p

KEYWORDS:
Drink water networks, hydraulic water nets modeling, water net revitalization, water net reliability.

ABSTRACT
The hydraulic calculations of sewage networks are done usually by the use of nomograms being the diagrams that show the relation between the main network parameters like pipe diameters, flow rates, hydraulic slopes and flow velocities. In traditional planning of sewage networks the appropriate hydraulic values are read mechanically from the the nomograms. Another way of calculation is the use of professional programs like the SWMM5 hydraulic model and genetic or heuristic optimization algorithms. In the paper another way of realizing the hydraulic and planning calculations is presented. The numerical solutions of nonlinear equations describing the main phenomena of sewage flows and a rather simple costs function are used in the paper to solve the tasks of hydraulic and optimization calculation of sewage networks.

INTRODUCTION
Modelling and planning of municipal sewage networks is a complex task because of the complexity of the equations describing the sewage flows in the canals. The basic hydraulic parameters describing a sewage net are sewage flows and sewage filling heights in the canals that result from the canal diameters and canal slopes. The standard approach of planning the sewage nets consists in using the nomograms which are the diagrams showing the relations between canal diameters, sewage flows, canal hydraulic slopes and flow velocities. The values of these variables are picked off from the diagrams that are results of the former calculation of the standard hydraulic formulas for computing the sewage network canals. A more advanced approach of planning the sewage networks bases on the use of professional programs for calculating the network hydraulic models like SWMM5 developed by EPA (US Environmental Protection Agency). The first approach with the diagrams is pure mechanical and the second one is rather complicated.

In the paper an indirect approach to calculate the hydraulic parameters of sewage networks is presented and it consists in relative simple numerical solutions of the nonlinear equations resulting from the basic hydraulic rules and formulas. The method proposed for modelling the sewage networks enables to analyse quickly the network parameters what makes it similar to the nomograms approach and it enables to understand easily the mutual relations between the different hydraulic parameters of the network canals. In this way this makes the method similar to the more complicated approach using the professional hydraulic models. Extending the formulas for hydraulic by a simple algebraic costs function and some limiting conditions, a very simple optimization problem for sewage networks is formulated also, that is quick and very easy to solve.

STATIC OPTIMIZATION OF SEWAGE NETWORKS
In the following considerations concerning the sewage networks optimization two kinds of canal slope are significant and they are the border slope and the slope causing the canal self-cleaning.

Border slope
Liquid flows with the fluid water level can be of quiet, critical or turbulent character. This depends on the value of the following Froude number [1, 4, 9]:

\[ F_r = \frac{v}{\sqrt{g \frac{A}{B}}} \]  

(1)

with: \( F_r \) – Froude number, \( v \) – average flow velocity [m/s], \( A \) – field of surface of active cross-section [m²], \( B \) – latitude of sewage level [m], \( g \) – gravity acceleration [m/s²].

Depending on the Froude number the sewage flow is quiet (laminar) \( (F_r<1) \) or critical \( (F_r=1) \) or turbulent \( (F_r>1) \). In the stationary flows with the free sewage level the slope of the canal bottom decides about the average flow velocity. Assuming the equality between the hydraulic slope and the canal bottom the critical slope has got the form [7]:

\[ J_{kr} = \frac{g \cdot U \cdot n^2}{\alpha \cdot B \cdot R^{\frac{1}{3}}} \]  

(2)

with: \( J_{kr} \) – critical slope [%], \( U \) – length of the canal circumference, \( R \) – canal hydraulic radius, \( \alpha \) – Coriolis coefficient, \( n \) – roughness.

The critical slope of a canal is a function of the canal geometrical dimension and the canal filling. The canal filling and circumference length are rising together monotonically and the level latitude and hydraulic radius have got their extremes. For a canal with the circle cross-section the level latitude is growing from zero value to its maximum point by filling half the canal and then draining it back to zero. The hydraulic radius is rising from zero to its maximum reached by the canal filling equal to 81,3 % and then it diminishes to the value reached by half the canal height. It results from it that the critical slope has got an extremum and depending on the filling degree \( x = H/d \) it is as follows:

for \( x \leq 0.5 \)

\[ J_{kr} = \frac{0.794 \cdot g \cdot n^2 \cdot \varphi}{\alpha \cdot \sin(0.5 \varphi) \cdot \left( \frac{\varphi - \sin(\varphi)}{\varphi} \right)^{\frac{1}{3}} \cdot d^{\frac{1}{3}}} \]  

(3a)

\( \varphi = 2 \cdot \arccos(1 - 2 \cdot x) \)  

(3b)

for \( x > 0.5 \)
\[ J_{kr} = \frac{1.587 \cdot g \cdot n^2 \cdot (\pi - 0.5 \cdot \varphi)^{\frac{3}{4}}}{\alpha \cdot \sin(0.5\varphi) \cdot (\pi - 0.5 \cdot \varphi + 0.5 \cdot \sin(\varphi))^{\frac{1}{2}} \cdot d^{\frac{1}{8}}} \]  
(4a)
\[ \varphi = 2 \cdot \arccos(2 \cdot x - 1) \]  
(4b)

with: \( H \) – height of the canal filling, \( d \) – canal diameter, \( \varphi \) – middle angle.

The diagrams showing the changes of the canal critical slope for different canal diameters depending on the canal filling degree are given in Fig. 1. The critical slope values for zero and full fillings are going to infinity. The critical slope \( J_{kr} \) reaches its minimal value for the canal filling equal to 29.7 \% and then it is known as border slope \( J_b \).

![Figure 1. Relations between the critical slope \( J_b \) and the filling degree \( x \) for different diameters \( d \).](image)

Assuming \( n=0.013 \), \( \alpha=1 \) and the filling degree \( x=29.7\% \) and after some transformations the following relation for the border slope results:

\[ J_b = \frac{3.778 \cdot 10^{-3}}{d^{1/3}} \]  
(5)

The sewage flows for the canal slopes smaller than border slope are always quiet; the canal fillings for the critical flows are called temporary fillings; the sewage flows proceeding between the temporary fillings are always turbulent flows.

Summarizing the above consideration one can say that the kind of sewage flow in a canal depends on the canal slope and canal filling and the particular significance has reached her the border slope \( J_b \). As it results from the Froude number the laminar flows arise only for the slopes less than border slope \( J_b \); for the canal slope equal to the border slope the flows are quiet with one exception of the temporary filling equal to 29.7\% when the flow is critical; for canal slopes bigger than the border slope and between the temporary fillings the flows are turbulent, they are critical for the temporary fillings and they are quiet for the remaining range of the filling values. It means that for securing the laminar sewage flow in a canal the canal slope has to be less than the border slope. There is to see from relation (5) that the border slope depends only on the canal diameter \( d \).

**Canal slope securing the process of canal self cleaning**

The sewage passing the canals shall have an appropriate big flow velocity called the self-cleaning speed. Such velocity in case of intensive sewage flows assures dilution and transport of the sediments that have been settled on the canal bottom at the time of smaller intensity of sewage flows. The self-cleaning speed can be secured when the friction between the sewage and canal wall is bigger than \( \tau_{\min} = 0.150 \text{ [kg/m}^2\text{]} \) for the rain wastewater and bigger than \( \tau_{\min} = 0.225 \text{ [kg/m}^2\text{]} \) for the communal and industrial sewage. [15].

The average pass tension between the canal wall and sewage is described with the formula [15, 9]:

\[ \tau = \rho \cdot R \cdot J \]  
(6)

with: \( \tau \) – pass tension [kg/m²], \( \rho \) – specific gravity of sewage [kg/m³], \( R \) – hydraulic radius for the canal filled partially [m], \( J \) – canal slope [%].

By means of relation (6) the minimal canal slopes \( J \) securing the self-cleaning speed by the gravitational sewage flow can be calculated:

for \( x \leq 0.5 \)

\[ J_{s} = \frac{4 \cdot \tau_{min} \cdot \varphi \cdot \ln \left( \frac{l}{d} \right)}{\rho \cdot (\pi - 0.5 \cdot \varphi) \cdot \varphi \cdot \ln \left( \frac{l}{d} \right)} \]  
(7a)

for \( x > 0.5 \)

\[ J_{s} = \frac{4 \cdot \tau_{min} \cdot \varphi \cdot (\pi - 0.5 \cdot \varphi) \cdot \ln \left( \frac{l}{d} \right)}{\rho \cdot (\pi - 0.5 \cdot \varphi) \cdot \varphi \cdot \ln \left( \frac{l}{d} \right)} \]  
(7b)

with: \( x=H/d \) canal filling degree, \( \varphi \) – middle angle calculated from (3b) or (4b).

The diagrams showing the change of minimal canal slope in dependence of canal filling degree for different canal diameters are shown in Fig. 2.

![Figure 2. Relations between the minimal canal slope \( J \) and the canal filling degree \( x \) for different canal diameters \( d \).](image)

From the diagrams results that the minimal slopes \( J \) shrink with the growing filling degrees what is fastest for the filling degree smaller than 10\% and that they reach the minimum by the filling degree equal to 60\% and then they are growing insignificantly. Inserting into the above formulas the value of hydraulic radius corresponding to the canal filling degree equal to \( x=60\% \) we get the following formula for the minimal slope ensuring the self-cleaning process of the canal:

\[ J_{s} = \frac{\tau_{\min}}{0.2776 \cdot \rho \cdot d} \]  
(8)

The diagrams describing the dependence of canal minimal slope \( J_{s} \) on the canal diameters \( d \) for different pass tensions \( \tau_{\min} \) are shown in Fig. 3. The slope values are diminishing with the growing diameter values \( d \).
In the following these limitations assuring the laminar sewage flows in the canals as well as self-cleaning process occurring there will be formulated.

**Limitations**

The first group of limitations has as aim to assure in the canals that the process of self-cleaning and the laminar sewage flow occur. These demands are fulfilled with the determination of specified canal slope that shall be bigger than minimal slope $J_s$ and less than the border slope $J_e$. Assuming in (5) and (8) the parameter values $n=0.013$ \([/m^{0.3}]/[s]^1\], $g=9.8067/[m]/[s]^2\], $\rho=999.6/[kg]/[m]^3\]$ and $0.25/[kg]/[m]^3\]$ these conditions can be written down in form of the following inequalities:

$$J > J_s \quad J - \frac{0.9 \cdot 10^{-3}}{d} > 0 \quad (9a)$$

$$J < J_e \quad J - \frac{3.778 \cdot 10^{-3}}{d^3} < 0 \quad (9b)$$

The second group of limitations cover the existence of the solutions of the nonlinear algebraic equations describing the relations between the canal filling degree $x$ and the sewage inflow $Q$ into the network in its stationary state of operation. By solving these equations the filling degree $x$ for the known network parameters and for the given sewage inflows $Q$ is stated. An detailed analysis of the equations is done in [2] and [8]. As results from there the solution of the equations exists when the following inequalities are fulfilled:

$$2\pi \cdot \beta > Q \quad \beta = 0.5 \cdot \frac{1}{n} \cdot (cd)^{8} \cdot \left(\frac{1}{4}\right)^\frac{5}{3} \cdot J^{\frac{1}{2}}$$

After some transformation the following relation is valid:

$$23,976 \cdot d^{8} \cdot J^{\frac{1}{2}} - Q > 0 \quad (10)$$

The dependence of parameter $\beta$ on the canal diameter $d$ and the canal slope $J$ is shown in Fig. 5.

**Optimizations of the Sewage Networks**

The problem discussed in the paper covers the gravitational sewage network divided into segments by the nodes. The nodes are the points of connection of several network segments or branches or the points of changing the network parameters as well as of location of sewage inflows into the network (sink basins, rain inlets, connecting basins). The investigated network covers the sanitary canalization operating in the conditions of stationary state. The optimization problem concerns the case when some new segments of the network are added to the existing ones. The optimization task is then brought to determine the canal slopes $J$ and canal diameters $d$ which by the minimal costs of building the new network branches can fulfill some limitations assuring their right operation. By the task solution the known sewage flows supplying the network in its nodes is assumed.
\[
F_2 (x) = 2 \cdot \left( \frac{(\pi - 0.5 \cdot \varphi_2 (x) + 0.5 \cdot \sin(\varphi_2 (x)))}{(\pi - 0.5 \cdot \varphi_2 (x))^3} \right) \quad (11)
\]

with:
\[
\varphi_2 (x) = 2 \cdot \arccos(2 \cdot x - 1)
\]

By the calculation of (11) for \( x = 70\% \) the following calculation results:
\[
5.26 \cdot \beta \geq Q \quad 20,0735 \cdot d^3 \cdot J_i \frac{1}{2} \geq Q \quad (12)
\]

**Costs function**

The costs of building the canalisation net depends of the following factors:

- network length (number of pipes)
- canal diameters
- deepening of the canals
- kind of the material from which the canals are made.

The right deepening of the canals shall assure:

- gravitational sewage inflow/outflow to/from the network
- sufficient covering of the canals regarding surface freezing
- avoiding the potential collision with the networks of other kinds and with their objects.

In order to fulfil these conditions the assumptions are done that for the sanitary networks the deepening \( h_m = 2.5 \) [m] and that the ground surface is flat. The costs function formulated for the optimization task consists of the cost of the canal deepening excavation and of the price of canal material. This function depends on the canal diameter, canal slope, canal length and canal deepening. Assuming the known values for the canal length and the deepening of the starting canal pipes \( L \) the following relation is valid:
\[
F(d, L) = \frac{1}{2} c_1 \cdot L \cdot \left( 2d^2 + d(2h_m + L \cdot J) + d \cdot \sqrt{d^2 + d(2h_m + L \cdot J)} + h_m \cdot L \cdot J \right) + c_2 \cdot L \cdot d \quad (13)
\]

with: \( L \) – length of the canal pipe [m], \( d \) – canal diameter [m], \( J \) – canal slope \([\%]\), \( h_m \) – minimal canal deepening on the begin of the canal [m], \( c_1 \) – costs of excavation of 1 m\(^3\) of the soil [\(\text{€/m}^3\)], \( c_2 \) – price of 1 m of the canal with the diameter \( d \) [\(\text{€/m}\)]

In the following, the problem of building the network branch consisting of \( K \) pipes is solved. The calculation task consists in defining the values of diameters \( d \) and canal slopes \( J \) that could secure the right operation of the network by the smallest costs of building this branch.

The network structure, the number and length of the pipes and the deepening of the beginning pipes are known. Then the optimization problem means the solution of the following task
\[
\min_{d, J} \left\{ \hat{F}(d, J) = \sum_{i=1}^{K} F_i (d_i, J_i) \right\} \quad (14)
\]

with the following limitations for \( i=1, \ldots, K \):
\[
J_i - \frac{3.778 \cdot 10^{-3}}{d_i} > 0 \quad (15a)
\]
\[
J_i - \frac{0.9 \cdot 10^{-3}}{d_i} < 0 \quad (15b)
\]
\[
20,0735 \cdot d_i^3 \cdot J_i \frac{1}{2} \geq Q_i \quad (15c)
\]

From the equation of the mass balance results:
\[
Q_i = \sum_{j=i} q_j
\]

and the following condition must be fulfilled for the pipe nodes which are connected with the neighbour pipes:
\[
h_{mi+j} = h_{mi} + L_i \cdot J_i
\]

where: \( q_i \) – sewage inflows to the \( i \)-th pipe from the connected pipes, \( h_{mi}, h_{mi+j} \) – minimal deepening of the initial nodes of the \( i \)-th and \( i+1 \)-th pipes, respectively.

Regarding the forms of the costs function and the added limitations one can conclude that such optimization task has a solution. The solution means the calculation of the following variables: canal slopes \( J \), canal diameters \( d \), minimal deepening of the initial pipe nodes \( h_m \), pipe lengths \( L \) and the sewage inflows \( Q \).

**EXAMPLE OF COMPUTATION**

The optimization task formulated above has been tested on the exemplary branch of the sanitary sewage network. The planned part of the network consists of 6 nodes connected to each other with the pipes. The network has got 3 input nodes (W6, W5, W3) and 1 output node W1. In the nodes W2 and W4 only the connection of some network pipes occurs (Fig. 6).
The initial data given at the beginning of the calculation or at the start of the calculation and describing the structure and the main parameters of the network planned are shown in Table 1. In Table 2 the results of optimization are placed.

**Table 1. The initial data of the network planned.**

<table>
<thead>
<tr>
<th>Upper node</th>
<th>Lower node</th>
<th>Segment</th>
<th>input flows in node $q$ [dm$^3$/s]</th>
<th>flows in segments $Q$ [dm$^3$/s]</th>
<th>Długość segmentu [m]</th>
</tr>
</thead>
<tbody>
<tr>
<td>W5</td>
<td>W4</td>
<td>1</td>
<td>0.11</td>
<td>0.11</td>
<td>55</td>
</tr>
<tr>
<td>W6</td>
<td>W4</td>
<td>2</td>
<td>0.32</td>
<td>0.32</td>
<td>100</td>
</tr>
<tr>
<td>W4</td>
<td>W2</td>
<td>3</td>
<td>0.22</td>
<td>0.66</td>
<td>42.5</td>
</tr>
<tr>
<td>W3</td>
<td>W2</td>
<td>4</td>
<td>0.24</td>
<td>0.24</td>
<td>30</td>
</tr>
<tr>
<td>W2</td>
<td>W1</td>
<td>5</td>
<td>1.13</td>
<td>2.76</td>
<td>276</td>
</tr>
</tbody>
</table>

**Table 2. Optimization results.**

<table>
<thead>
<tr>
<th>Segment</th>
<th>$J$ [%]</th>
<th>$d$ [m]</th>
<th>$H/d$ [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.55</td>
<td>0.199974</td>
<td>3.27</td>
</tr>
<tr>
<td>2</td>
<td>0.532</td>
<td>0.210293</td>
<td>8.27</td>
</tr>
<tr>
<td>3</td>
<td>0.531</td>
<td>0.212931</td>
<td>10.99</td>
</tr>
<tr>
<td>4</td>
<td>0.546</td>
<td>0.200033</td>
<td>5.99</td>
</tr>
<tr>
<td>5</td>
<td>0.515</td>
<td>0.214309</td>
<td>20.93</td>
</tr>
</tbody>
</table>

As results from Table 2 the differences by the calculated canal slopes $J$ and by the canal diameters $d$ are very small and they are practically negligible by all 5 network pipes. This is caused by the small values of the forecasted sewage inflows $q$. By these inflow values the canal filling degrees do not exceed 20% in all pipes and their values are placed always between the minimal slope $J_{min}$ and the border slope $J_{max}$.

The canal deepening values and the pipe lengths have no any impact on the values of the canal slopes and diameters but they influence the value of the costs function. If we assume for example that the cost of excavation of 1 m$^3$ of the soil is $c_1=12[\text{€}/m^3]$ and the price of 1 m length of the canal is $c_2=2[\text{€}/m]$ then the value of the costs function will be equal $F=3.520.53[\text{€}]$.

The network investigated was also calculated by means of the MOSKAN system worked out in IBS PAN [13] This IT system is based on the hydraulic model SWMM5 developed by EPA [11]. The results obtained while using the MOSKAN system are the very similar to these obtained while solving the above optimization task: for all network pipes the canal slopes have the values $J=0.51[\%]$ and their diameters are $d=0.2[m]$. It means that the algorithm of sewage network optimization presented in the paper is not worse regarding its exactness than the more complicated method applied in MOSKAN and using the SWMM5 hydraulic model and at the same time it is more simple and clear.

**References**

ENERGY
CONSUMPTION
SIMULATION
OPTIMAL LOAD SHARING STRATEGY IN SYSTEM OF MULTIPLE SOURCE BASED ON MODEL CONTROL PREDICTIVE

Mouhamadou Falilou NDIAYE  
Pape Alioune NDIAYE  
CIFRES – Ecole Supérieure Polytechnique UCAD  
Dakar, Sénégal BP 5085  
e-mail: mouhamadou.ndiaye@ucad.edu.sn,  
papaas.ndiaye@ucad.edu.sn

François GUERIN  
Dimitri LEFEBVRE  
Research Group in Electrotechnics and Automatics  
(GREAH) of the University of Le Havre, 75 rue Bellot, CS80540, 76058 Le Havre Cedex (France)  
e-mail: francois.guerin@univ-lehavre.fr,  
dimitri.lefebvre@univ-lehavre.fr

KEYWORDS
Generalized adaptative PID, Model control predictive, energy, multi source

ABSTRACT
This manuscript describes the control of the energy transfers in Systems of Multiple Sources of Energy (SMSE). The considered SMSE is a Linear Time Varying (LTV) system in view of the sources and load fluctuations. The proposed control scheme has two levels. The first is used to regulate the DC bus voltage and the current of each source. It is carried out by generalized adaptive PID controllers. The second level determines automatically the power rate associated with each source. The calculation of the power rate is computed with a Model Predictive Controller (MPC) designed to optimize the cost per kWh produced by the system. Simulations are performed using measurements taken over the village of Gandon in Senegal.

INTRODUCTION
For several years many works have concerned renewable energies (Chauhan and Saini, 2014) (Baños et al., 2011). Some of them proposed the improvement of the components performance: solar panels, wind turbine, energy storage, static converters (Benson and Magee, 2014) (Cameron and van der Zwaan, 2015). In particular, algorithms are developed for tracking maximum power point (Gadelovits et al., 2014). The sites that are not supplied by the power distribution network have an appropriate settlement for energy supply with SMSE. In a lot of cases it is more economically interesting to use renewable energy. Another advantage is the reduction of environmental impact by reduction of emission greenhouse gas. But the stochastic nature of renewable sources requires their combined use in order to increase energy availability. Several topologies are proposed with or without storage (Upadhyay and Sharma, 2014). Some authors proposed to optimize the sizing (Kotroutsos et al., 2006) or the management (Arul et al., 2015). The aim of the sizing optimization is to reduce the initial cost investment. In some cases load shedding is accepted. It can favorably impact on the initial investment cost. Sometimes a shedding less than 5% greatly affects on the initial cost investment (Bilal et al., 2013). Some authors have proposed to control the exploitation cost, to improve system reliability and efficiency (Baños et al., 2011) (Iqbal et al., 2014). Usual methodologies are: multi objective optimization, fuzzy logic (Kumar et al., 2014) (Guérin and LeFebvre, 2013), predictive control (Dieulot et al., 2015) (Q. et al., 2011) and neural network (Ciabattoni et al., 2013) (Alphousseyni et al., 2014). Furthermore, several works on hierarchical control have suggested the coupling and decoupling of sources. In this paper, a control scheme is proposed based on a hierarchical approach previously developed by the authors (Guérin et al., 2012) (Guérin and Lefebvre, 2013) (Guérin et al., 2011). Indeed, in their previous works the authors adjust the duty cycle value (DC/DC converter input control) because this variable correlates negatively the available energy and the energy required by the load. They used this property to automatically adjust the power ratio of each source thanks to a fuzzy logic coordinator (Guérin and Lefebvre, 2013) and they select the best operating modes with an automaton (Guérin et al., 2012).

The main contribution of this paper is the automatically calculation of the power ratio with
Model Predictive Control (MPC) in order to optimize the economic criteria respecting the technical restraints assigned. Dispatching is important and necessary for any electrical network especially in SMES. It ensures equality between production sources and electrical load.

This paper is organized as follows: in Section II the description of the considered system is presented. Section III explains the coupling modeling and local control of the hierarchical control scheme. This local control is achieved by generalized PID controller which regulates the DC bus voltage and the current of each source (Guérin and Lefebvre, 2013). The technical characteristics are also presented. The design of MPC controller witch compute optimal value is detailed in section IV. The formulation of the cost function based on the economic objective is also exposed in the same section. In section V the case of Gandon’s village is studied. Measurements made on the site are detailed as well as the results from simulations. Section VI is devoted to the conclusion and prospects. We made a summary and suggested a suitable pursuit of the work.

TOPOLOGY OF THE SMSE

SMSEs that are studied in this paper are made up of heterogeneous power sources coupled on a DC bus through DC/DC converters. The supervisory control has been developed (Guérin et al., 2012) for SMSEs with an arbitrary number $N$ of sources and illustrated for $N=3$ (Figure 1). For simplicity and portability, SMSEs with a conventional source (for example, a diesel generator, named source 1), a renewable source (for example, solar panels, named source 2) and a storage device (for example, a batteries bank, named source 3) will be considered. Such SMSEs cover a large variety of applications and supply variable loads in single-phase or three-phase alternative form (inverter). Sources 1 and 2 are connected on the DC bus via identical Zero Voltage Switching (ZVS) full bridge isolated Buck converters and source 3 is connected on the DC bus with a reversible Buck Boost DC/DC converter. In Figure 1, $I_{dc}$ stands for the current of source $i$, $S$ stands for the DC bus voltage, $V_{ci}$ stands for the DC/DC converter $i$ analog voltage control input and $X_{pi}$ stands for the logical value of the contactor that couples source $i$ on the DC bus. Gains $K_{v}$ will be defined in section III.B.

The considered SMSEs may run in several operating modes (Guérin et al., 2012). Each operating mode corresponds to a configuration that is characterized by the sources that are coupled on the DC bus. As a consequence, $2^N$ modes will be considered at most for a SMSE with $N$ sources (Guérin et al., 2011). This supervisory control selects the operating mode and has been fully described by the authors in (Guérin et al., 2012).

An automaton (global stage) selects the best operating mode of the SMSE according to several parameters such as the off periods for the renewable source, the state of charge value for the storage source, the charge sustain, and so on. When an operating mode is selected, the local stage regulates the DC bus voltage and the current provided by each source according to the power ratio provided by the coordinator. It’s important to note that the characteristics of the sources are not required for the proposed control design. In this paper, we propose a new coordination stage. It is based on the use of the MPC and has to compute the optimal values of the power ratio of each source starting from an accurate model which describes the coupling of the sources on the DC bus. To describe this new coordination stage, conventional and renewable sources are assumed to be coupled and battery is assumed to be fed (mode 6 in (Guérin et al., 2012) is considered).

LOCAL STAGE CONTROLLER

Modeling of sources coupling

The structural diagram of the ZVS full bridge isolated Buck converter is represented on the Figure 2. These DC/DC converters are isolated (HF transformer TR1) Buck converters ($D_5$, $D_6$, $D_7$, $D_8$, $L$, $C_e$, $R_e$) with a full bridge ($Q_1$, $Q_2$, $Q_3$, $Q_4$).
$Q_3$, $Q_4$ and ZVS. The full bridge control ($Q_1$, $Q_2$, $Q_3$, $Q_4$) is realized by a phase shift controller UC3879 through specialized MOSFET drivers IR2113. The duty cycle value $\varphi$ is modified by the phase shift between $V_a$ and $V_b$ voltages. The phase shift is controlled by an analog DC voltage (between 0V and 5V) which represents the DC/DC converter analog voltage control input ($V_{CC}$).

![Figure 2. Structural diagram of the ZVS full bridge isolated Buck converter](image)

State, control and output vectors are respectively defined as $X=\begin{bmatrix}I_{L1} & I_{L2} & S\end{bmatrix}^T$, $U_k=\begin{bmatrix}V_{CI} & V_{C2}\end{bmatrix}^T$, $Y_k=\begin{bmatrix}I_{L1} & I_{L2}\end{bmatrix}^T$. The average state space model is written with equation (1):

\[
\begin{align*}
\dot{X}_k &= A_kX_k + B_kU_k \\
Y_k &= C_kX_k \\
U_k &= D_kX_k
\end{align*}
\]

Note that in comparison with our previous works (François Guérin and Lefebvre 2013), (F. Guérin et al. 2011), this average state space model has been simplified. It does not take into account the primary resistance ($r_p$), the secondary resistance ($r_s$) and the magnetizing inductance ($L_m$) of the HF transformer (TR1) as well as the MOSFET transistors ($Q_1$, $Q_2$, $Q_3$, $Q_4$) channel resistance ($r_{mos}$). These parameters can be neglected due to their very low values and the estimation of the global losses is not required.

$L$, $r_L$, $C_e$ and $R_e$ are respectively the coil inductance, the coil resistance, the capacity and the resistance of the Buck converter. $n$ is the ratio of the HF transformer. $V_{CI}$, $V_{C2}$ are the DC/DC converter analog voltage control inputs (0V/5V). $V_{CI}$, $V_{C2}$ are the source voltages. $K_{PS}$ is the gain of the phase shift controller UC3879. The duty cycle value is proportional to the analog voltage control input so that $\varphi_3=1$ when $V_{CI}=5V$ ($\varphi_3=K_{PS}V_{CI}$). $R_{EQ}$ is the load (supposed to be resistive). It depends on the current provided to the consumers ($I_{con}$) supplied through an inverter and eventually to the current used to charge the batteries bank ($I_{bb}$). Note that a current regulator is used to charge the batteries bank so the current $I_{bb}$ is supposed constant. As a consequence, $R_{EQ}$ can be modeled as $R_{EQ}=S/(I_{con}+I_{bb})$. $R_{EQ}$ is the equivalent resistive load on the DC bus: $R_{EQ}=(R_{c1}+R_{c2})/(R_{c1}+R_{c2}+R_{bb})$. $C_{dc}$ is the DC bus capacity. $C_{EQ}$ is the equivalent capacitor on the DC bus: $C_{EQ}=2Ce+Cdc$. $\mu_i$, $\mu_f$ are respectively the gains of the current sensors (LEM-LA55P) and the voltage sensor (LEM-LV25P) associated with their signal conditioners.

**DC bus voltage and current regulation loops**

The considered SMSE is a Linear Time Varying (LTV) system in view of the sources and load fluctuations. For this reason, adaptive generalized PID controllers have been investigated to control the DC/DC converters under varying operating conditions (Guérin and Lefebvre, 2013). The goal is to keep for the voltage and current loops (local stage) the same dynamic behaviour (without overshoot or oscillations) whatever the sources and load fluctuations, when the power ratio is modified by the coordination stage.

Under the constraint that the DC bus voltage is constant, it is possible to drive the power delivered to the load by controlling the current provided by each source (load sharing). The cascaded controller (Figure 3) has been designed starting from the average state space model (1) and has been fully described in (Guérin and Lefebvre, 2013). It regulates the DC bus voltage (whatever the sources and load fluctuations) and the current provided by each source. It also includes adjustable (between 0% and 100%) gains ($K_i$, $K_d$) making it possible to control the power ratio provided by each source.

![Figure 3. Local stage controller](image)

The dynamic behavior of each loop (current and voltage) can be described (Guerin et al., 2012) with first order closed loop transfer functions according to equations (2) to (4):

\[
I_L(s) = \frac{1/\mu_f}{1 + \frac{1}{H_iK_{C2}\mu_f}} \cdot I_{PWR}(s) = \frac{1/\mu_i}{1 + \frac{1}{H_iK_{C1}\mu_i}} \cdot I_{PWR}(s)
\]
\[ I_{r1}(s) = \frac{1/\mu_i}{1 + s/H_1K_{r1}\mu_i} \]
\[ I_{r2}(s) = \frac{1/\mu_{r2}}{1 + s/K_{r2}S_{ref}} \]
\[ S(s) = \frac{1/\mu_i}{1 + s/K_{r1}}S_{ref} \]
\[ I_{ref1}(s) = K_{r1}I_{ref2}(s) \]
\[ I_{ref2}(s) = K_{r2}I_{ref2}(s) \]
\[ I_{ref}(s) = \frac{\mu_i(1 + \tau_r s)}{R_{eq}} \]

with: \( K_{r1} + K_{r2} = 1 \) and \( K_{r1} > 0, K_{r2} > 0 \)

\( I_{ref1}, I_{ref2} \) and \( S_{ref} \) are respectively the reference values of the current and voltage loops. \( \tau_r \) is the desired time constant for the current loops. \( \tau_v \) is the desired time constant for the voltage loop (DC bus). Note that we neglect the loss of the ZVS full bridge DC/DC converters, the duty cycle values \( (\theta_1, \theta_2) \) can be easily estimated (Guerin et al., 2012) thanks to the following equations:

\[ S(s) = n_{g2}(s)V_{g2}(s) \]
\[ I_{ref}(s) = n_{g2}(s)V_{g2}(s) \]

In equation (7), the output voltage \( V_o \) is the same for both DC/DC converters since they are coupled on the same DC bus. To conclude, the proposed cascaded adaptive controller does not have steady state errors (for both currents and DC bus voltage) and will keep the desired dynamic behavior \( (\tau_r, \tau_v) \) whatever the sources and load fluctuations when the power ratios \( (K_{r1}, K_{r2}) \) will be modified by the coordination stage (load sharing).

**COORDINATION STAGE CONTROLLER**

The stage of coordination is based on Model Predictive Control Approach (MPC). The MPC paradigm uses an internal model for predicting the system behavior and choose the best decision according to a cost criteria, while respecting the functional constraints (Richalet et al., 2004). MPC is widely used in the field of energy (Baños et al., 2011). In the field of SMSE, it is already used to optimize production (Qi et al., 2011) and monitoring control issues (Kaabeche and Ibitiouen, 2014), (Qi et al., 2011). Our approach is to use an economical cost function that will minimize the cost per kWh produced. This function is based on the calculation of hourly cost of each source. The power available from a renewable source is calculated from its technical characteristics and from the meteorological data. Its knowledge is very important as it affect substantially the cost per kW and determines the production restraints.

**Design of the internal model**

The MPC controller (coordination stage) has to compute the optimal values of the power ratios \( (K_{r1}, K_{r2}) \). To do that, an internal model is designed in which the power ratios \( (K_{r1}, K_{r2}) \) belong to the input vector and the duty cycle values \( (\theta_1, \theta_2) \) current belong to the output vector. Over the horizon of prediction, we assume that the DC bus voltage \( S \) is constant and equal to \( S_{ref} \) (the voltage regulation loop does not have steady state error). Thus, equation (6) can be rewritten according to (8):

\[ \mu_i S_{ref}(s) = R_{eq}I_{ref}(s) \]

Starting from equations (2) to (8), the internal model can be written according to equation (9).

\[ \dot{x} = f(x, u) \]

\[ y = c(x, u) \]

In discrete time with sampling period \( T_e \), equation (9) is rewritten according to a first order approximation:

\[ X_{k+1} = F X_k + G U_k \]

\[ Y_k = C \cdot X_k + D U_k \]

with:

\[ F = I + A \cdot T_e \] and \( G = B \cdot T_e \)

\( I \) is the identity matrix of appropriate dimensions.

**Cost criteria**

The previous internal model is used to predict the behavior of the system and choose the best decision according to a cost criterion while respecting the functional constraints.

For each source \( i \), the cost function takes into account the annualized costs of acquisition \( (C_{ai}) \), maintenance \( (C_{mi}) \), renewal \( (C_{ri}) \) and hourly operation cost \( (C_{oi}) \) in the forecast horizon. These costs are computed from the initial costs. Details are described in (Kaabeche and Ibitiouen, 2014):
\[ C_i^l = \frac{C_i}{(1 + \tau)^D} \]  

(12)

\[ C_i^m = \frac{C_i^M}{D} \]  

(13)

\[ C_i^r = \frac{C_i}{(1 + \tau)^D} - 1 \]  

(14)

with:

- \( C_i^l \) : cost of acquiring source \( i \)
- \( C_i^r \) : cost of replacing source \( i \)
- \( C_i^m \) : cost of maintenance source \( i \)
- \( D \) : life span
- \( \tau \) : inflation rate

The hourly cost of the system is given according to equation (15) (Mbojdi et al., 2014):

\[ C_i^h = \frac{C_i^l + C_i^r + C_i^m}{8760} \]  

(15)

where 8760 is the number of hours in a year and \( P_i \) is the power produced by source \( i \). The total cost of the system at time \( k + n \) is the sum of the costs of each source:

\[ J_{k+n} = \sum_j J_{j,k+n} \]  

with:

\[ J_{j,k+n} = \frac{C_{i_j} + C_{i_{j+k+n}}}{8760} \cdot P_{j,k+n} \cdot \Delta t \]  

(17)

\[ C_{i_{j+k+n}} \] operating cost per hour of source \( i \) at date \( k+n \):

\[ C_{i_{j+k+n}} = 0 \]  

(16)

\( C_i \) : rate of the liter of fuel

\( P_{j,k+n} \) and \( P_{j+k+n} \) are respectively the power produced by the SMSEs and the source 2. They are rewritten with (18), (19):

\[ P_{j,k+n} = S_{j,k+n} \cdot (I_{1,j,k+n} + I_{2,j,k+n}) = Y_{j,k+n} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]  

\[ P_{j+k+n} = S_{j+k+n} \cdot (I_{1,j+k+n} + I_{2,j+k+n}) = Y_{j+k+n} \cdot \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \]  

(16) results in (20):

\[ J_{k+n} = \left( \begin{bmatrix} Y_{k,n}^T & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \cdot \begin{bmatrix} k_1 \end{bmatrix} \right) \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} P_{n} \end{bmatrix} \begin{bmatrix} 0 & 0 \end{bmatrix} \begin{bmatrix} k_2 \end{bmatrix} \]  

(20)

\[ \text{subject to:} \]

\[ y_{j,k+n} = \sum_j Y_{j,k+n} \cdot Y_{j+k+n} \cdot \begin{bmatrix} \frac{C_i + C_i^l + C_i^r + C_i^m}{8760} \cdot \Delta t \end{bmatrix} \]  

(21)

The total cost over the prediction horizon \( H \) is given by (22):

\[ J = \sum_{m=1}^{H} \sum_{j=1}^{I} U_{j,m} \cdot H \cdot k_j + \sum_{j=1}^{I} f_{j+k,m} \cdot U_k \]  

(22)

Let us assume that the sources, the load, and the control vector remain constant over the horizon of prediction, the system output \( Y \) (9) is predicted over \( H \) steps with the equations:

\[ X_{j,m} = F \cdot X_{j,m} + \sum_{i=1}^{I} F_i \cdot Y_{j,i,m} \]  

(23)

\[ Y_{j,m} = C \cdot F \cdot X_{j,m} + \left( \begin{bmatrix} C \cdot \left( \sum_{i=1}^{I} F_i \right) \cdot G + D \end{bmatrix} \right) \cdot U_j \]  

The cost criterion \( J \) is minimized (with respect to \( U_j \)) by using a standard quadratic optimization method. The following constraints are required by the load sharing and the available power of each source:

\[ K_j + K_{j+H} = 1, \quad 0 < K_j, K_{j+H} < P_{j+H}^m \]  

(18)

\[ C_{m} \] the average cost per kWh produced by \( 8 \text{MSE} \) are given according equations (24)

\[ C_{m} = \frac{1}{P_{j+H} \cdot dt} \int_{0}^{P_{j+H}} dt \]  

(24)

**Mathematical model of available power**

The available power of the conventional source is constant and equals its nominal value. The
wind power and photovoltaic available are detailed by (Kaabache and Ibtioen, 2014) (Belfkira et al., 2011). They are given according equations (30) (31)
\[
\begin{align*}
P_{vol} &= \frac{\alpha}{a} P_{n,vol} \quad \text{if} \quad v_c < v < v_n \\
0 &\quad \text{if} \quad v < v_c \text{ or } v > v_n \\
\end{align*}
\]
(25)
\[
\alpha = \frac{P_{n,vol}}{v_c - v_n} \quad \text{and} \quad b = \frac{v_c^3}{v_n - v_c}
\]
P_n,vol: rate power of the wind turbine (W)
P_{vol}: available power of the wind turbine (W)
V_c: starting speed of wind (m/s)
V: wind speed at the projected height h (m/s)
V_n: nominal speed of wind
V_c: Cutting speed
P_{PV} = I_{mpp} \cdot V_{mm}
I_{mpp} = I_{sc} \left[ \left(1 - \lambda_1 \exp \left( \frac{V_{max}}{\lambda_2 \cdot V_{oc}} \right) \right) \right] \cdot A I
V_{mpp} = V_{max} + \mu_{v,oc} \cdot A I 
\]
(26)
P_{mpp} = V_{mpp} \cdot I_{mpp}
\lambda_1 = \left(1 - \frac{I_{max}}{I_{sc}}\right) \cdot \exp \left( - \frac{V_{max}}{\lambda_2 \cdot V_{oc}} \right)
\lambda_2 = \left(\frac{V_{max}}{V_{oc}} - 1\right) \left[\ln\left(1 - \frac{I_{max}}{I_{sc}}\right)\right]^{-1}
\Delta I = I_{sc} \cdot \left(\frac{G_T}{G_{ref}} - 1\right) + \mu_{v,oc} \cdot A I
\Delta T = T_C - T_{C,ref}
T_C = \frac{NOCT - 20}{800} \cdot G_T

With:
P_{PV}: available power of solar panel (W)
I_{sc}: short-circuit current
T_C: cell operating temperature
T_{C,ref}: PV panel temperature at reference operating conditions is equal to 25°C
T_A: ambient temperature of the site under consideration (°C)
V_{mpp}: voltage at maximum power point
I_{mpp}: current at maximum power point
NOCT: normal operating temperature
V_{max}: maximum voltage of PV panel at the reference operating conditions (V)
I_{MAX}: maximum current of PV panel at the reference operating conditions (A)
V_{oc}: open circuit voltage of PV panel (V)
\mu_{v,oc}: Temperature coefficient for PV panel (V)

SIMULATIONS AND RESULTS

To highlight the performances of the controllers (local and coordination stages), several Matlab/Simulink simulations have been done with the following parameters: \(L=120\mu \text{H} , r_l=0.5\Omega , \ C_{el}=330\mu \text{F} , R_w=56K\Omega , C_{dc}=1\mu \text{F} , n=10, \mu_r=1V/A, \mu_i=6.25mV/V, K_{ps}=0.2V^{-1}\). In addition, \(T_s=0.15\text{ms}\) is the sampling period. For the current and voltage regulation loops the desired time constants are respectively \(\tau_r=20\text{ms}, \tau_i=200\text{ms}\). Sources’ characteristics are: \(P_{n,vol}=500\text{W}, V_{ref}=3\text{ms}^{-1}, V_{oc}=7\text{ms}^{-1}, V_{ref}=10\text{ms}^{-1}, \text{ISC}=3.33\text{A}, V_{max}=17.4\text{V}, I_{MAX}=3.11\text{A}, V_{DC}=21.7\text{V}, \mu_{v,oc}=-8.21*10^{-3} \text{V/°C}, \mu_{v,sc}=1.33*10^{-3} \text{A/°C}\). Number of solar panel and wind turbine are respectively 182 and 10. P_{2n}, 8kW D=20 years and \(r=3\%/\), \(C_{in}=27.7k\epsilon, C_{out}=2.27k\epsilon, C_{ref}=1.2\epsilon\).

Noted \(S_{ref}=300V\) is the desired DC bus voltage measurements of meteorological variables are recorded in table 1. The load profile and the calculated available renewable power from the equation 2 are shown in figure 5.

The source voltages \(V_{G1}, V_{G2}\) are represented in Figure 6. They vary between the maximum \(120\text{V}\) and minimal \(30\text{V}\) values tolerated by the DC/DC converter. The adaptive generalized PID adjust the duty cycles values (Figure 7) in order to regulate the DC bus voltage (Figure 8) and the current provided to the load by each source (Figure 9). One can notice that the variations of the power ratio and the source voltages do not disturb the regulation loops.

The MPC calculates the optimal power ratios \(K_1\) and \(K_2\) (Figure 10). The power ratios are automatically calculated and adjusted at each time step. The sum of the source currents (Figure 9) is equal to the current required by the load (Figure 11). MPC controller is integrated perfectly with the low-level control. Despite the variation of the voltage sources bus voltage is maintained constant and equal to the desired value: 300 V. It is also in accordance with the supervisory control scheme proposed by previous work (Guerin et al., 2012). Indeed at date 2h-15mn, 3h-15mn, 6h-30mn, 16h45mn, and 22h-15 we observe a mode change.

In this work we simulated the situation where
the source voltages vary in normal operating conditions. Sometimes the source voltage 2 is the double of the source voltage 1 (for example, at time instant 4h). We can see that the MPC controller calculates the best power ratios in order to keep the lowest energy cost.

![Figure 4. Load profile and availability of renewable power](image)

![Figure 5. Voltage of the sources](image)

![Figure 6. Duty cycle](image)

![Figure 7. Currents of the sources](image)

![Figure 8. DC bus Voltage](image)

![Figure 9. Rate of power](image)

![Figure 10. Currents Load](image)

![Figure 11. Cost per kWh produced](image)

**Table 1: Load and measured weather data**

<table>
<thead>
<tr>
<th>Hour (h)</th>
<th>Wind speed (m/s)</th>
<th>Temperature(°C)</th>
<th>Irradiance (W/m²)</th>
<th>Load (kW)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>4.177</td>
<td>0</td>
<td>24.99</td>
<td>1.6</td>
</tr>
<tr>
<td>1</td>
<td>3.852</td>
<td>0</td>
<td>24.76</td>
<td>1.5</td>
</tr>
<tr>
<td>2</td>
<td>3.852</td>
<td>0</td>
<td>24.21</td>
<td>1.3</td>
</tr>
<tr>
<td>3</td>
<td>3.852</td>
<td>0</td>
<td>23.052</td>
<td>1.2</td>
</tr>
<tr>
<td>4</td>
<td>4.052</td>
<td>0</td>
<td>22.452</td>
<td>1.3</td>
</tr>
<tr>
<td>5</td>
<td>4.355</td>
<td>0</td>
<td>22.028</td>
<td>1.1</td>
</tr>
<tr>
<td>6</td>
<td>4.558</td>
<td>0</td>
<td>21.587</td>
<td>1.6</td>
</tr>
<tr>
<td>7</td>
<td>3.912</td>
<td>0.00133</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>8</td>
<td>5.795</td>
<td>21.063</td>
<td>21.063</td>
<td>2.8</td>
</tr>
<tr>
<td>9</td>
<td>6.773</td>
<td>21.063</td>
<td>21.063</td>
<td>2.8</td>
</tr>
<tr>
<td>10</td>
<td>7.324</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>11</td>
<td>7.987</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>12</td>
<td>8.176</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>13</td>
<td>7.405</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>14</td>
<td>8.251</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>15</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>16</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>17</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>18</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>19</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>20</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>21</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>22</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>23</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
<tr>
<td>24</td>
<td>6.826</td>
<td>21.063</td>
<td>21.063</td>
<td>2.9</td>
</tr>
</tbody>
</table>

**CONCLUSION AND PERSPECTIVES**

In this paper, a control scheme that includes adaptive generalized PID and model predictive control has been proposed to improve the load sharing in SMSEs. The local controllers cooperate with a MPC coordinator that tunes the power ratio provided by each source according to the load and duty cycle estimations. Compared to our previous work (Guérin and Lefebvre, 2013), the MPC coordinator compute the optimal values of the power ratios by taking into account economical considerations and functional constraints. We can note that use meteorological data to estimate the available energy. In our future works, both controllers (local stage and coordination stage) will be implemented into a Microchip dsPIC30F6010A microcontroller with industrial DC/DC converters. The meteorological data will be obtained online weather station. We intend to
use a GSM modem. We will also improve the cost criteria to take into account technical performance. Our main efforts will finally be devoted on the supervised control with automata in order to find the optimal configurations according to the climatic conditions.

REFERENCES
WIRELESS SENSOR NETWORK LIVE CIRCLE SIMULATION TOOLS TO BALANCE ENERGY CONSUMPTION IN THE NETWORK NODES

Aleksejs Jurenoks
Leonids Nocickis
Riga Technical University
Riga, Latvia
E-mail: aleksejs.jurenoks@rtu.lv, lnoviciks@gmail.com

KEYWORDS
Wireless sensor network, life circle assessment, network power consumption

ABSTRACT
Nowadays such sectors as agricultural, industrial, and private have used for years embedded information and processing technologies that are morally outdated and cannot be adapted to today's requirements. The integration of new wireless sensor network technologies can improve the management of existing infrastructure, enabling real-time control and analysis of information received from monitored devices. Despite the high costs of the system, this technology can be justified by the introduction of the solution from the initial stages. Nowadays, the most topical research pertaining to wireless sensor networks are grounded on the new optimization of structure of network transmission protocol, the routing optimization in transmission network, optimization of network structure, as a result of which the life circle of wireless network sensors is possible to increase. There are a lot of ready-made solutions that allow to accomplish the simulation process for wireless sensor networks operating mode. As the essential disadvantage of the relevant wireless sensor network simulation tool can be considered non-integration of the methods for the optimization of wireless sensor networks lifetime. The integrated methods in simulation systems use inefficient network optimization methods for the design of optimal wireless sensor network systems. These include the choice of the capacity of individual battery, the deployment of the node density, the adjustment of power transmitter, the application of energy-efficient data transfer protocol, positioning of network nodes and other methods that are associated with the introduction of additional network costs. This article will discuss the simulation tool for the assessment of a wireless sensor network life circle that ensures the configuration of optimal network and projection/displaying of topology operations by using energy balancing methods.

INTRODUCTION
Sensor networks are primarily intended for the collection of information. This means that in any network there are terminal nodes for collection of information, the router nodes for transmission of information, and one or more coordinator nodes on which all the collected information across the network is sent. The nodes pertaining to information storage and processing are mainly equipped with an uninterrupted power supply, independent distribution network connectivity and high performance computing systems. This means that the direction of the data is definitely known in the sensor network, the information from the data collectors is sent to the information storage and processing nodes by using existing routers of the network. Transmitting information in the network, the capacity of data traffic near coordinator node is much higher than in the distant points. The differentiated energy consumption in hubs [Комаров and Воско 2012] always appears in operating sensor network; as a result, the existing elements close to processing nodes faster than others stop operating due to lack of electricity and, as a result, the network ceases its overall work. The necessity for modeling of wireless sensor network simulation is based on several factors:

- Inability to perform physical experiments using existing technology, existing solutions do not allow fully carry out wireless sensor network lifetime optimization methods.
- It is necessary to carry out the experiment for longer time interval to create in the network the large unbalance of the remaining energy. It will allow to investigate how the system will be able to make the necessary changes in the network topology for energy balancing.
- Despite the variety of network performance modeling tools offered on the market, the wireless sensor network simulation process requires the tool that supports:
  - The support of the dynamic network reconconfiguration;
  - The task support for the optimization of the network structure;
  - The change of network properties and the graphical monitoring in the mode of emulation process.

The article describes the developed wireless sensor network simulation software tool for the determination of optimal network configuration.

RELATED WORKS
Other authors have described the simulation tools pertaining to wireless sensor network process system that allows to determine the life circle of the network by using different energy balancing methods. The architecture of the heterogeneous network requires different methods to improve energy efficiency.

1. The capacity of the battery in the adaptation based on the location of the element in the network [Gun et al. 2007]. In this case, the nodes, which transmit more information, are equipped with a more powerful battery to ensure a smooth power consumption across the network. This
approach is easy to implement in life, but it has significant drawbacks - low network scalability and poor adaptation of functioning change. Moreover, the technical specification and design solutions of the various nodes significantly raise the cost of the network.

2. The differentiated density of network node is dependent on the estimated transmission density in a specific area [Hald et al. 2011]. The aim of this solution is to create redundancy in the network structure and provide a separate node duplication. As a result, in the case of denial of one of the routers, its functions will be separated to another node duplication which until that moment was not used and was in sleep mode with small power consumption.

At the level of simulation tool software there are mainly used routing protocols based on the remaining power units nodes [Chen 2006], the use of virtual coordinates [Bakasos 2009], distant and close turn of transmission [Zhang et al. 2009], node positioning [Komaros and Bocson 2012] as well as the use of clusters [Soro et al. 2005].

It is known that in the protocols of traditional network router metrics are used to increase network transmission capacity and prevent data transmission delays. The functions of the metrics may perform various intermediate network nodes to the final aim, the communication channel capacity, the line load level [Olimfer and Olimfer 2010]. The functions of the metrics to the final aim, the communication channel capacity, the line load level may perform various intermediate network nodes [Olimfer and Olimfer 2010]. In the sensor networks there are often used node remaining energy metrics to track the data processing unit. In this case, from different alternative routes will be selected one, on which nodes there is greater amount of the remaining energy.

This article will discuss the simulation tool for the assessment of a wireless sensor network life circle that ensures the configuration of optimal network and displaying of topology operations by using energy balancing methods [Jurenoks and Novickis 2015]. In this article described wireless sensor network simulation tool allows to assess each individual element of the network life circle by using wireless sensor network primary activity types. It allows to assess the overall life circle of the network taking into account the options of network reconfiguration.

**MATHEMATICAL MODEL OF WIRELESS SENSOR NETWORKS**

Sensor networks can be defined as the amount of mathematically oriented graph $G$, where each graph $G$ consists of vertices $V = \{1, 2, ..., m\}$, edges $U: U \subseteq V \times V$ and total energy consumed by graph in all $P$ nodes.

$$G = \{V, U, P\}. \quad (1)$$

$V_s$ indicates to a potential graph in the position, in which the network may be located. The condition of the graph in our case is determined by the location of the coordinator in the segment of the network.

In the real situation in the wireless sensor networks the coordinator node can occupy any position in the network graph. There are situations when the position of the next location is possible to predict, but there are such situations when a position is selected at random. As a result, the possible condition of the graph can be expressed by the following formula:

$$|V_s| = \left( \begin{array}{c} m \\ s \end{array} \right) = \frac{m!}{s!(m-s)!}. \quad (2)$$

The graph theory is widely applied in the simulation tools that ensure a large-sized network modeling process. Some widely used simulation tools [Jurenoks and Novickis 2015] apply offered by manufacturers energy balancing method using a differential transmitting power. [Zhang et al. 2009]

It is assumed that each network node can operate in two modes - close transmission mode and the long-range transmission mode. Working in the close transmission mode, the node passes the information to their nearest neighbor, working in the long transmission mode, the node transmits information by reducing the amount of re-transmission in the network. If the network has $n$ nodes than it can be assumed that each node, except the coordinator node, can choose the operating mode independently of other nodes. As a result, the following amount of configuration is possible:

$$|V_s| = 2^n \quad (3)$$

Applying the formula 3., it can be seen that the amount of configuration depends upon the amount of nodes in the network. Using dynamic reconfigurable network, this type of addictions do not exist. The graph of $G_s$ groups forms the wireless network $G_s(n) = (V_s, E_n)$, $k \in V_s$, where $V_s$ – the amount of vertex, $E_n \subseteq V_s \times V_s$ – the amount of edges. Peaks correspond to the network nodes, the amount of edges make the connection of the channel quantity in the network.

Each node of the sensor network $v_i = (E_i, P_i, S_{P_i})$ is characterized by the initial amount of the energy $E_i$, the amount of consumed power in a set $P_i = (p_{i1}, p_{i2}, ..., p_{iP})$ where $p_{ik}$ – is the consumed power by $i$ nodes while performing $k$ activity and there is the $S_{P_i}$ node/s $i$ power list, where $F_i^\prime$ is the additional energy that is needed for $i$ node transition from configuration $j$ to configuration $k$.

The initial energy of network coordinator is considered to be unlimited and can be expressed by the formula:

$$P_i = (E_i \rightarrow \infty). \quad (4)$$

On the basis of the formula nr. 4, it can be considered that consumed power by coordinator node does not affect the total life circle of the wireless sensor network; thus, the integration of the simulation in the process is not needed.

In this article, the proposed simulation tool can be used only in networks with strict definition of node properties, which are expressed on a constant energy consumption for each node element being set in a concrete operating state.

**EVALUATION INDICATORS OF LIFE CIRCLE PERTAINING TO WIRELESS SENSOR NETWORKS**

The life expectancy of the network depends on how long its elements are in operation. Taking into account that the networks very often contain verbiage, there are different requirements for the quality of the results obtained from the network. Carrying out the network simulation process, it is important to evaluate the precise operation time of each
network element until the battery of the element will require replacement. Any sensor network has three node types - terminals, routers, and data collectors. Let us assume that the data collectors do not affect the overall life circle of the system because they are provided with independent power supply or are equipped with a much powerful autonomous power supply. Currently in wireless sensor network life circle assessment simulation tools there are widely used universal indicators for determination of life expectancy.

The Indicator Based on the Working Node

The total time of network operation can be marked as \(T_k^n\) time, in which at least \(k\) of the \(n\) units are in operation [Chen et al. 2001, Kang et al. 2003, Vass et al. 2005]. However, there is one drawback of this labeling/marking- the types of network nodes are not defined. Mostly in the networks there are primary nodes - which provide data retransmission and significantly impact the overall network performance. If one of the "bearing" nodes stops working, it automatically means that the network ceases to exist. That is why in many works [Helmman et al. 2006, Raiciu et al. 2005], analyzing the life circle of system, variable \(m\), which means the number of critical important elements in the network, which must be active at all times, is defined. For example they could be the nodes of network cluster management [Soro et al. 2005]. For the other nodes, metrics is used:

\[
T_k^n = m\min T_i. 
\]  

(5)

In the researches it is often used a case when \(k = n\). In this case the network is considered as able-bodied as long as all the network nodes are active or:

\[
T_n^n = \min T_i, 
\]  

(6)

where: \(V_n\) - the quantity of network nodes; \(T_i\) - the life circle of each network node.

The Indicator Based on the Coverage of Area

This indicator is related to the use of network sensor- the quantity of information that is necessary to get from the concrete network segment. There are two approaches on the determination of the indicators based in the coverage that are used for network modeling tools:

1. The network is considered as able-bodied as long as a percent of the overall network coverage is covered by at least one sensor (\(\alpha\)-coverage) [We et al. 2005].
2. The second approach is based on provision of redundancy and requires that in each network segment should be at least \(k\) - number of active sensors (\(k\)-coverage) [Kumar et al. 2004].

The biggest disadvantage pertaining to the detection of indicators based coverage is the complex process of algorithmization.

Indicator Based on the Delay of Transmission of Information

According to the research [3GPP 2008, 3GPP 2009, C. Mehlf et al. 2009, Chen 2006], this indicator is mostly used when working with sensor network systems pertaining to information gathering. The full definition of the indicator is provided in the paper [Chen 2006]. Let us improve this indicator in order to cover previously mentioned indicator groups.

Suppose that the environment, which is carried out by the sensor network monitoring system, at each time interval occurs in certain activities which need to be identified. The events can be considered as external events, such as alerts, temperature changes of the system and other technical information, as well as described determinate event of the internal network, such as a regular transmission of information.

Let us introduce the parameters of quality pertaining to working network for each zone at specified time interval \(t\). Suppose that \(N_k(t)\) - in the range \(k\) there is the total amount of events in the time interval \(t\) and \(I_k(t)\) - the total amount of the events from the number \(N_k(t)\) that are delivered to a data collector at the determined period of time. The total time \(t\) is defined for the whole range \(k\). The parameter \(\Delta t_k\) is defined on the basis of the necessary network intensity in the determined network segment. In the result zone at \(k\) time in the \(t\) interval the indicator of network working quality will be equal to:

\[
Q_k(t) = \begin{cases} 
\frac{I_k(t)}{N_k(t)}, & \text{if } N_k(t) \neq 0 \\
1, & \text{if } N_k(t) = 0.
\end{cases} 
\]  

(7)

For acting network there is a threshold \(c_k\) that exists in order to indicate the lowest ratio value \(Q_k\) below which the network can not be regarded as working age. Thus, for the possible duration of the network can be considered:

\[
\forall t < \tau_k, \forall k \in [1..m]: Q_k(t) \geq c_k. 
\]  

(8)

PROCESS SIMULATION ALGORITHM FOR EVALUATION OF WIRELESS SENSOR LIFE CIRCLE IN SIMULATION TOOL

Network life circle depends on the selected network topology, which is determined by the place where network coordinator is situated. Suppose that the coordinator of the network is mobile and the motion path (route) network – \(S\) of the coordinator is known. In the result the modified sensor network model by introducing coordinator route parameters – \(N = (V_a, U_n, S, P)\) will be used. Assume that the network uses the same amount of energy in each of the nodes, further in the article we will assume that at the beginning the amount of energy in a node \(E_i=\)P. The primary task is to calculate what the network node energy balance will be, when the coordinator will gather information from the \(l\) network nodes:

\[
E_r(l) = E_i - (\sum_{m=1}^{l} P_m). 
\]  

(9)

where \(E_r(l)\) - the amount of remaining energy of node \(l\) after the evaluation of \(i\) amount of nodes. The evaluation of wireless sensor network life circle depends on the network operating schemes, the movement type of the coordinator and the time when coordinator is in each position. The network will cease to operate when at least in one the nodes the remaining amount of energy \(E_r \leq 0\).
Let us consider an algorithm (Figure 1) that allows to determine the maximum number of iterations of the network until at least one of the nodes will stop work. The result algorithm indicates the maximum number of iterations that a network can perform provided that the data is collected every time the coordinator node activates in the network or changes the position in the network and node identifier with the low amount of energy.

**Figures 1:** Algorithm for detection of the maximum number of interactions pertaining to network

long as all the elements are active $T^n = \min_{i \in V_n} T_i$ will be detected by using Figure 2 of the algorithm. While network is in operation, there is time when it is necessary to make the network topology change. Topology change can be carried out in the changed position of the coordinator in the network and the network agents which activities are described in [Bakka2009]. Network life circle with moving coordinator will be determined by existing algorithm provided by Figure 3.

**Figures 2:** Algorithm for detection of network life circle without changes of network topology

**Figures 3:** Algorithm for detection of network life circle with change of network topology

When algorithm is executed to detect the life circle, the coordinator at the beginning is placed in $k$ position. The starting amount of energy for all the nodes is defined in wireless sensor network model $E_{ij} = P$. The network operates using the generated topology as long as remaining amount of energy in each node is not less than the average amount of energy in the network $E_{ij} < \text{average}(E_i)$. If necessary, the network carries out topology change by placing a coordinator in the position, where the network can operate as long as possible without changing the topology.

**WIRELESS SENSOR NETWORK LIFE CIRCLE ASSESSMENT MODELLING TOOLS**

The quality of the results pertaining to network modeling depends on the accuracy of exit information that is defined at the beginning of the modeling stage of the system. In other authors’ works [Bassam and Raouf 2006, Ping et.al. 2005, Soro et.al. 2005] containing the description of the modeling system, the description of simulated network is carried out through the description of the separate components or completed defined blocks (e.g., router, terminal, etc.). The ready-made solution for the use of wireless sensor network emulating is not efficient because:

- There is the excessive complexity for description of each element under consideration in the network.
Wireless sensor network uses the same type of network elements. On the basis of carried out researches [Gun et al. 2007], the maximum size of network which can be emulated by using the approach mentioned comprises 104 objects in the network. When the number of objects under consideration exceeds the maximum limit, the system is unable to carry out the emulation process.

- It is impossible to describe the structure of information transmission pertaining to network and algorithms. To make network emulation coordinator level, it is necessary to define each connection path between objects. It is not possible to automatically generate the optimum transmission route based on defined conditions.
- It is not possible during emulation process to change the network topology and to use dynamic structure. Making changes to the network structure or changing one location of element, the modeling system in the network will allow to make automatic reconfiguration and continue the emulation process.

Table 1 provides the comparative analysis of the currently widely used wireless network simulation systems with the aim to show their compatibility with simulation of operating wireless sensor networks.

### Table 1: Comparative analysis of Wireless Sensor Network Simulation Modeling systems

<table>
<thead>
<tr>
<th>Requirements</th>
<th>NS-2</th>
<th>NS-3</th>
<th>OPNET</th>
<th>Odette++</th>
<th>Anylogic</th>
<th>SIMULA</th>
</tr>
</thead>
<tbody>
<tr>
<td>The level of detail</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Graphical interface</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Representation of modeling process</td>
<td>+</td>
<td>+</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Necessary information storage</td>
<td>-</td>
<td>-</td>
<td>+</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Modelling of WAN network</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Accounting for the time factor</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
<tr>
<td>Network topology change during modelling process</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>+</td>
</tr>
</tbody>
</table>

The network simulation method described in the article use tool with the structure provided in the Figure 4.

**Figures 4: Structure of wireless network process simulation tool**

The core software is responsible for the three basic functions of the system:
- Ensures the planning of network nodes in the particular territory;
- Network topology formation on the basis of the given conditions;
- The control of the remaining energy while the system is in operation.

The graphical interface of the system provides the display of network topology and the remaining amount of energy in the nodes. The software performs modelable model life-cycle simulation and points out the weaker points of the system, which reduces the life circle of the overall system.

Using the system described in the article, there is an opportunity to simulate all the possible standards of wireless sensor networks and network modules. Using the network model which supports the ZigBee protocol is the possibility to modify ow-level IEEE 802.15.4 model code; as a result, the common network operating scheme can not be modified. Wireless sensor network model supports three possible network topologies: star, tree, cellular network. The model includes three types of nodes (according to the ZigBee standard) - coordinator node (Coordinator), the router node (router) and the terminal unit (End Device).

In the software the sensor node has four functional levels:
- At the physical level it is possible to control the transmission parameters - wtx and receiving parameters - wrx, which in accordance with IEEE standards 802.15.x can use 868 Mhz, 915 Mhz and 2.4 Ghz frequencies. The properties of physical network are defined at the level of coordinator. The concealed power of transmitter in the system = 5mW.
- Environmentally-level approach is implemented by CSMA / CA algorithm according to the standard IEEE 802.15.4.
- The network level is implemented in accordance with the ZigBee specification.
- At the application level the range of functions are realized in accordance with the ZigBee specification. Software WIRELESS SENSOR NETWORK model does not support the following functions:
  - Broadband data transmission (broadcast).
  - Security options.
  - Modification of CSMA / CA centralized transmission standby.

**CONCLUSION**

The developed wireless sensor network simulation tool allows to assess each individual element of the network life cycle by using wireless sensor network node primary activity types. It allows to assess the overall life circle of the network taking into account the options of network reconfiguration. The most important difference of the systems from previously developed tools includes the wide application of the system and adaptive surveying network features: network topology, wireless standards, the technical characteristics of the module, the operating frequency, topology change. Performing simulation using the described in the article modeling tool, enables to assess the efficiency of alternative wireless sensor network increasing life circle methods and algorithms [Osis et.al. 2008], as well as to determine the network configuration requirements to be met by wireless sensor network during modeling time. The offered tool allows modeling in the most effective way, as long as possible for the network to be in the autonomous action, and makes the necessary technical requirements involved for each network module.
During the process of piloting the tool described in the article, there might be highlighted the following individual cases:

- The solution offered in the article was used in the framework of the EU project Unite, which was implemented in cooperation with Riga State Gymnasium Nr. 3, during the period 01.02.2006 - 01.07.2008. In the course of the project the monitoring tool of the learners’ load was designed.

- The optimization modeling methodology pertaining to information transmission was approbated in the EU project E-logimarM, which was implemented during the period 2004 – 2006. In the course of the project the concept of the optimal path for the network was set up in order to guide monitoring of load by using sensor nodes.

- In cooperation with CCFI funds in 2012 the autonomous energy-efficient "Smart House" maintenance and management of wireless sensor network has been designed and used; moreover, it is integrated in 12 private houses in Latvia. The developed technical solution has extended the interval of home sensor module service from industrially defined 2 years to 5 years.

REFERENCES


Communications in Computer and Information Science 22, 387 - 398.


Олифир В. Г., Олифир Н. А. Компьютерные сети. Принципы, технологии, протоколы. // СПб.: Питер, 4-е издание, 2010. 943 с.

WEB REFERENCES


SIMULATION OF CHANGE POWER DEMAND IN MICRO-GRID SUPPLIED BY MIX OF SOURCES INCLUDING AGGREGATE OF SMALL-SCALE HYDROPOWER PLANTS

Eugeniusz M. Sroczan
Poznan University of Technology
Institute of Electric Power Engineering
ul. Piotrowo 3A, 61-249 Poznań, Poland
e-mail: eugeniusz.sroczan@put.poznan.pl

KEYWORDS
electrical engineering, decision support system, interactive simulation, optimisation, smart-grids systems, prosumer

ABSTRACT
The DSS destined for a real-time operation of the committing hydro-power plants and the thermal units supplied has been developed. The results of simulation possibly improve the rated coefficient of the use of the primary energy of renewable sources.
The optimisation of load dispatch considers two aspects: the water management policy of hydro-power plants and the variability of electricity demanded by prosumers.
The proposed algorithm of the optimisation of the efficiency of primary energy conversion and minimisation of the cost of electricity includes the hierarchy of applied primary and secondary criteria.

INTRODUCTION
Decentralization and automation of micro-grid control cannot worsen the stability of the electricity grid system (EGS). A changeability of the value of demanded power generated by renewable sources is a main cause of possible interferences.
Factors that influence the EGS working in the quantic of the multidimensional vector \( \{ H, O, S, T \} \) are deterministic and stochastic.
The following factors were divided into individual groups:
H – describes managing of hydropower plants (HPP) reservoirs, preference of hydroelectric power stations, at present also wind ones (WPP), correlation of costs of the energy from HPP and WPP with fuel costs in thermal power plants;
O – takes into account the environmental factors and policy of renewable energy utilization;
S – elasticity of the transmission-distribution system;
T – free or limited access of the third party (TPA) to the grid system.
The vector \( \{ \text{HOST} \} \) is determined for the given configuration of committing plants, as a subset of EGS devices and takes into consideration the possibility of the optimisation of the cost of generated energy as well micro-grid, as in EGS.
The structure of HOST vector is described in the paper (Sroczan 2014).
The primary kind of energy, which is the aim of this paper, is extended by wind power plants and small scale hydro-

power plants which are committed in smart grid (SG) or micro-grid (MG) (Pijarski et al. 2014). Electricity generated in such mixed aggregate of power plant depends on several conditions.
The policy of saving energy from renewable sources of energy (RES) for the production of electricity is expected to be integrated in the electricity systems (Klos, Paska 2013, Dąbrowsk, Hunik 2015, Sroczan 2014). It is known that cost of electricity raises due to the integration of RES on the performance of the grid system operation. Integration leads the merit order to shift to relatively low variable costs, especially wind (WPP), photovoltaic power plants (PVPP) (Krawczyk, Serrano 2015), and small scale hydropower plants (SHPP).

Some SHPP located in series on the river create the cascade of power plants. Therefore, the kind of operation of the SHP causes the possible usage of hydro-energy which is converted to electricity. The main aim of this paper states as follows: the development of a simple method of definition and modelling of the essence of phenomena effecting in the efficiency factor of energy conversion in hydropower plant in mixed micro-grid.

PURPOSE OF SIMULATION
Modelling of power demand in micro-grid
The characteristics of micro-grid (MG) is a bidirectional flow of energy and information between the energy user and the utility grid allows energy users not only to consume energy, but also to generate the energy and share the excess.

![Figure 1: Simulated Characteristics of Load Caused in MG](image)

This is the main reason for paying higher attention to the research on the investigation of the MG-related aspects of
prosumers. That is a technical and essential point of view on the problem of the management of the integration of power system. The power generated by renewable sources is not sufficient to cover the load. Therefore, the remaining part of the power is being balanced with EGS energy. In the fig. 1, the simulated load variability of two sets of end users $P_{\text{c}}$ and $P_{\text{e}}$ supplied from MG sources: WPP, PVP and HPP is shown. The value of $P_{\text{MG-P2G}}$ describes the variability of balancing power. The tool of simulation of demanded power load was described in (Sroczan, Urbaniak 2013).

Simulation of water flood in hydropower plants cascade

Equations (1) and (2) describe the model of change of the upper and lower level of water surface, which is suitable for simulation:

$$Z_{\text{oi}}(t) = Z_{\text{li}}(t) + V_{\text{oi}} \left[ \int_{0}^{t} h_{i}(\tau) \cdot Q_{\text{oi}}(T - \tau) d\tau \right] - Q_{\text{oi}}(t) dt$$

$$Z_{\text{oi}}(t) = Z_{\text{bi}}(t - \tau) + \lambda_{i} \Delta Z_{\text{oi}}(t) + \left[ \Delta Z_{\text{oi}}(t)(1 - \lambda_{i}) \cdot \text{exp} \left( \frac{-\tau}{T_{i}} \right) \right]$$

where:
- $V_{\text{oi}}$ [m³] – the initial storage of the upper reservoir of the hydropower plant; $V_{\text{i}}$ [m³] – the storage of the reservoir in the moment $t$;
- $h_{i}$ (m) – flood function for the given $Q_{i}$ (t) impetuous impact
- $\Delta V_{\text{oi}}$ – the sum of leakage of the reservoir, $i$ – i-th power plant; $T_{i}$ – period of calculation; $t$ – time; $\tau$ – time increment;
- $Q_{\text{oi}}$, $Q_{\text{ti}}$ [m³/s] – the inflow and outflow of water of $i$-th plant;
- $Z_{\text{oi}}$ – the momentary level of water surface of lower reservoir; $\Delta Z_{\text{oi}}$ – change of water level surface for the given water flow calculated for steady state of flow;
- $\lambda_{i}$ – initial rise in level of water surface of lower reservoir.

It is assumed that other limits are also fulfilled:
- Water level limits of reservoir:
  $$Z_{\text{D}_{i}}(t) \geq Z_{\text{D}_{i}}(t) \leq Z_{\text{U}_{i}}(t) \leq Z_{\text{U}_{i}}(t)$$

where: $Z_{\text{D}_{i}}$, $Z_{\text{U}_{i}}$, $Z_{\text{U}_{i}}$, $Z_{\text{U}_{i}}$ [m] – the minimum and maximum water levels of the reservoirs
- Utilization of water at required downstream limits of water flow:
  $$Q_{\text{oi}}(t) \geq Q_{\text{oi}}(t)$$

$$Q_{\text{ti}}(t) \leq Q_{\text{ti}}(t)$$

where: $Q_{\text{oi}}$ – the minimum discharge capacity of the i-th hydropower plant for downstream ecological requirements, restricted by the downstream flood control limitations; $Q_{\text{ti}}$ [m³/s] – the maximum discharge capacity of the i-th hydropower plant;
- Power generation limits caused by technical and/or grid system operator:
  $$P_{\text{g}_{i}}(t) \leq P_{\text{g}_{i}}(t) \leq P_{\text{g}_{i}}(t)$$

where
- $P_{\text{g}_{i}}$ [kW] – the output of the i-th hydropower plant in the moment $t$;
- $P_{\text{g}_{i}}$ [max] [kW] – the maximum power capacity of i-th hydropower plant.

The set of characteristics of water consumption per unit (as shown in fig. 2.) is applied to calculate the optimal power dispatch among the sources supplying the MG using the relation:

$$\eta = \frac{336.97}{h \cdot v_{i}}$$

Figure 2: Measured and Calculated Characteristic of Water Consumption for Hydropower Plant HPP 1; at Different Head, where $H1$ > $H2$. (Kuziemski 2011)

The characteristic of water consumption per unit enables to decide how the plant should generate the electricity to obtain maximum profit. There are two attempts: work with the maximal value of spill causing the maximal volume of electricity using the disposable volume of water or frame work with maximal power, the volume of electricity decreases in this case.

Simulated cost of electricity generation

The procedure of simulation of the cost of energy generated takes into account the wind power plants and a cascade of small-scale hydropower plants. It is assumed that two hydropower plants cooperate as a cascade located on the river. The first plant in series is named disposable, the next works as a control of flood levels, however, both can significantly influence the power generating efficiency. The inflow of water for the intermediate plant is calculated using the relation (1).

The cost of the generation for a given set of sources is calculated by determination of the power load $P_{\text{SgG}}$ generated by EGS units, which is defined as follows:

$$P_{\text{SgG}} = \sum_{i} P_{\text{MG}_{i}} - \sum_{i} P_{\text{MG}_{j}}$$

where: $P_{\text{MG}_{i}}$ [kW] – demanded power of each $i$ user in the time $t$, $P_{\text{MG}_{j}}$ [kW] – generated power load of each $j$ plant committing in the MG. The value of the $P_{\text{SgG}}$ power (1) is obtained from EGS sources balancing the current power demanded in MG and generated by MG sources. The committing sources of EGS generate the power:

$$P_{\text{SgG}} = \sum_{i} P_{\text{SgG}_{i}}$$

In the operational research $\text{T}$, the summary cost of the generated energy is calculated from the relation(8) for sources of the transmission system (EGS) and for similar MG sources:

$$C_{\text{g}} = \text{min} \left[ \sum_{i} C_{i}(P_{\text{g}_{i}}) \right]$$

$k \cdot P_{\text{g}_{\min}} \leq P_{\text{g}_{i}} \leq P_{\text{g}_{\max}}$; $k = 0 \lor k = 1$

$C_{i}(P_{\text{g}_{i}})$ – cost of generation in $i$-th unit of EGS, $C_{i}(P_{\text{g}_{i}})$ – cost of generation in $j$-th unit of MG, $P_{\text{g}_{i}}$, $P_{\text{g}_{i}}$ – the range of
load of committing units as well for MG as for EGS, 
\( k \) – switch on/off defining the possibility of shut down the 
given unit, for example set of HPP.

**Cost optimization with regard to state of MG**
This decision making process concerns the balance of the 
power demanded in MG. The manager of the power system 
decides on the power value which must be generated. The 
operation cost of the generation of electricity due to differ-
et kind of power plants is not the same. This comes from 
two main reasons: decisions are made for short period (in 
range of 15 to 30 minutes) and cost depends on the current 
state of the power system.

Taking into consideration the relations (6), (7), (8), (9) and 
(10), the economical power dispatch (EPD) between MG 
power stations (units) fulfills the relation:

\[
\gamma \frac{\partial C_H}{\partial W} = \gamma \frac{\partial C_W}{\partial P} = \lambda = \min \left( \frac{\partial C}{\partial P} \right) 
\]

(11)

where the increment of the cost of generated energy is de-
defined as: \( \gamma \frac{\partial C_H}{\partial W} \) – for hydro-power plant; \( \gamma \frac{\partial C_W}{\partial P} \) – for wind 
turbines and \( \lambda \frac{\partial C}{\partial P} \) – for thermal power plants.

The decision is made with regard to two criteria: balance the 
power and minimize the cost of the generation of electricity. 
Therefore, it is necessary to declare which of them is the 
primary one (Sroczan 2014).

**SCOPE OF SIMULATION**
In the developed simulator, the set of energy consumers is 
represented by stated EU groups (energy user), each of rated 
power \( P_{EUR} \). Result of this simulation is described in fig. 1. 
Simulation tools for integration management processes of 
dispersed generation in micro-grid contain:

- economic power dispatch (EPD) on the level of MG,
- home energy management or energy management sys-
tem (HES, EMS),
- demand side management (DMS),
- maximal utilisation of possible to convert energy from 
  hydropower plants as result of simulated water inflow 
  to HPP and outflow.

Set of varying power demand is obtained from the require-
ments stated by behavior of end users (prosumers) of 
electricity. The second step of optimization (EPD) enables to 
calculate the optimal load of each source, committing in 
MG and EGS.

**RESULTS OF SIMULATION**
Figure 1 presents the demand load as a result of simulation. 
Taking into consideration the changeability of load in the 
time of operational research (in this case – a week), a graph 
tidied up of demanded power for energy user is obtained. 
The simulation of the demanded power allows to determine 
the influence of the load to the system cost in terms of reser-
vation of the power and purchase/sale of the energy betwee-
en the smart grid and power system with respect to the rules 
of economy. The result is shown in the figure 3. After the 
modification of the manager’s decision, using relation (7) 
the power plant TPP1 is loaded as the first one and TPP2 as 
the next one.

![Figure 3: The Family of Characteristics of EPD Among Power Units in MG Calculated for Values: \( \gamma = 0.1 \) \( \kappa = 0.1 \)](image)

The assessed impact of the PS manager on the strategy of 
the load of the HPP and WPP depends on the value of \( \gamma \) the system equivalent price of water which is consumed in each 
hydro-power plant and similarly price of wind \( \kappa \). For HPP, 
an additional effect is due to value of water consumption, 
which results from the head of water. That value depends 
on the state of the work in the preceding period.

The cost of generation calculated for each power plant 
shows that for low \( \gamma \) power plant TPP1 is load as a first one. 
If the value of \( \gamma \) increases, the load is covered by power 
plant TPP2 (Fig. 3). It means that in the first case the cost of 
emission of combustion gases is lower than in the second 
one.

**CONCLUSIONS**
The aim of the simulator worked out is defined as the opti-
misation of the generation cost with respect to primary en-
ergy savings and EGS constraints.
The described simulation process includes the MG as a sub-
et of EGS in order to minimize the costs of energy with re-
spect to the expectations of the end-users of energy and 
rules defined for EMS (energy management system).
The structure of simulator based on modified fuzzy dynamic 
programming and special process of modifying the ener-
getic profile of prosumers gives an universal tool supporting 
the management of large scale system using very simple 
and reliable tools optimizing the cost of the generated 
energy. Especially MG contains a set of hydropower plants.

**REFERENCES**
Dąbrowsk J., Hutnik E. 2015. „Profitability analysis of energy 
production from a micro installation of renewable energy in a 
Klos M., Faska J., 2013. „Methodology of economic assessment of 
Pijsarski P., Rzepecki A., Wydra M. 2014. „Efficient management 
Krawczyk D.A., Serrano A. R., 2015. „The usage of solar energy 
for electricity production on example of Andalusia (Spain)”. 
Kuziński J. 2011. [“Improvement the hydrotechnical and energy 
parameters of hydroelectric power stations”]. MSc thesis. 
Poznan University of Technology.
Sroczan E. M. 2014. „Simulation tools for integration manage-
ment processes of dispersed generation in micro-grid supplied by pro-
sumers”. In Brito A.C et al., (eds), *Proc. of ESM2014 Conf. 
ESM’2013 EUROSISETI Publ.*, Ostend, Belgium, p. 358-360.
Sroczan E. M., Urbaniak A. 2013. „Simulation of processes of 
electricity generation integrated in smart grid model manage-
ment”. In Onggo S., Kavieka A. (eds.), *Proc. of ESM 2013 
EUROSISETI Publication*, Ostend, Belgium, p. 142-146.
REAL-TIME TASK RECONFIGURATION IN ENERGY-HARVESTING BASED MULTIPROCESSOR SYSTEMS

Wiem Housseyni, Olfa Mosbah, Mohamed Khalgui
National Institute of Applied Sciences and Technology,
Institute-University of Carthago, Tunisia
Email: {wiem.housseyni,olfamosbah,khalgui.mohamed}@gmail.com

Maryline Chetto
University of Nantes IRCCyN UMR CNRS
6597 44321 Nantes, France
Email:maryline.chetto@univ-nantes.fr

KEYWORDS

ABSTRACT
This paper deals with real-time scheduling in a reconfigurable multiprocessor system. Each processor is supplied with a renewable energy source, and uses a rechargeable energy storage. A reconfiguration scenario is any operation that consists in the addition, removal or update of tasks which may result in timing unfeasibility. In our work, each task is represented by a Probabilistic Directed Acyclic Graph (DAGP). Our contribution in this paper concerns the online feasibility problem that is issued from the reconfiguration process. We describe new approaches to solve this problem: the first one changes the timing parameters of the DAGP, the second one decomposes and migrates branches of the DAGP, the third one modifies the scheduling mode, and the fourth one deletes some tasks.

INTRODUCTION
Distributed embedded systems are widely used in real-time domains such as biomedicine, automobile, aircraft and industrial areas. In real-time systems, not only the treatments should be correct but they have to produce outputs in bounded time (Stankovic 1988). For soft real-time systems, missing some deadlines may be tolerable. In contrast, for hard real-time systems, non respect of timing requirements may lead to severe consequences (Liu and Layland 1973, Burns 1991, Manacher 1967).

Batteries are the dominant energy source for embedded real-time systems. However, in addition to their negative impact on the environment, their use may be troublesome due to their limited energy storage capacity and their finite useful live. For some application domains, replacing battery is either costly or impractical. Hence, these embedded systems should be designed to operate incessantly and be autonomous through renewable energy sources. Several technologies are proposed for environmental energy harvesting, in particular solar and vibrational energies. Energy harvesting emerges as a promising technology to surmount energy limitation and enhance system lifetime. In an energy harvesting based system, the main issue is to guarantee that at any moment the system does not consume more energy than harvested and available in the storage.

In this paper, we consider an energy harvesting based real-time system which is composed of a set of identical processors. Preemption and migration of tasks are authorized on every processor. The hardware platform is supplied with renewable energy sources. The harvested energy is stored in rechargeable energy reservoirs with limited capacity (e.g. batteries / supercapacitors).

A reconfigurable computing system undergoes unpredictable events that require adequate online decisions so as to maintain schedulability of the application software. Reconfiguration should be performed whenever a task needs to be added, removed or replaced or a task needs to modify its timing parameters for applicative motivations. The problem of real-time task scheduling on a reconfigurable multiprocessor architecture has received substantial attention (George et al. 2005, Camponogara et al. 2010, Gharssellaoui et al. 2012). In contrast, only few works deal with multiprocessor counterparts. Furthermore, considerable studies consider the problem of minimizing energy consumption so as to maximize system autonomy (Wang et al. 2015, Chniter et al. 2014). But they do not address the energy neutrality problem that characterizes energy harvesting based computing systems.

Prior works on real-time scheduling in energy harvesting systems addressed basic models where tasks are independent from each other. And they mainly focus on monoprocessor architectures even with DVFS facilities. However, these results cannot be simply extended to tasks which are represented by directed acyclic graphs (DAGs) in a distributed context. Consequently, both static task assignment and dynamic task migration in multiprocessor energy harvesting systems are new challenges. We will address these two issues. Firstly, we
will contribute with proposition of a more realistic task model based on DAG. Secondly, we will provide new solutions for task assignment and dynamic reconfiguration. In our model, every task \( \tau_i \) can be viewed at two distinct levels: i) at the first one, \( \tau_i \) is a computation box with given period \( T_i \), worst case execution time \( C_i \), relative deadline \( D_i \) and worst case energy consumption \( E_{\text{peak}_i} \); ii) at the second level, \( \tau_i \) is represented by a probabilistic directed acyclic graph called Probabilistic DAG (DAGP). The real-time simulator Cheddar (Singhoff et al. 2004) permits to verify the systems behavior and evaluate the performance of our different approaches. Simulation results bring to light the effectiveness of the proposed DAGP model and reconfiguration strategies measured in terms of deadline miss ratio and energy savings.

The remainder of the paper is as follows. Section II gives a brief state of the art about both energy aware scheduling and directed acyclic graph based models. Section III formalizes the scheduling problem. Section IV presents the new task model, namely Probabilistic DAG (DAGP) and the terminology used throughout this paper. Section V describes our solutions for reconfiguration in real-time energy harvesting based multiprocessor systems. Section VI reports simulation results that bring to light the effectiveness of these solutions measured in terms of deadline miss ratio. And finally, we conclude in Section VII with a summary of our contributions.

RELATED WORKS

In this section, we present a state of the art successively about real-time scheduling under energy harvesting constraints, reconfigurable real-time systems and scheduling of DAG tasks.

Real-time Scheduling and Energy Harvesting Considerations

For about ten years, several important works have focused on scheduling in uniprocessor energy harvesting based embedded systems. In (Moser et al. 2006), the Lazy Scheduling Algorithm (LSA), based on the EDF (Earliest Deadline First) rule is proved to be optimal for periodic or aperiodic tasks with deadlines. In (Chetto 2014), another optimal scheduling algorithm, called EDH is proposed, based on the EDF rule, with less restrictive hypotheses than LSA. Energy harvesting aware scheduling for the multiprocessor case has received much less attention. Among the most interesting studies, Abdallah et al in (Abdallah et al. 2014) describe and evaluate real-time task assignment heuristics for optimizing the global deadline success ratio. All of these studies consider independent and modular task execution models.

Reconfigurable Real-Time Systems

Several interesting academic and industrial works focused on reconfigurable systems where automatic reconfigurations are applied by intelligent agents (Khalgui 2010). More recently, studies dealt with the same issues with low-power considerations. In (Chen et al. 2014), two combinatorial optimization approaches based on DVFS processors are described for minimizing energy consumption. A mechanism adjusts deadlines so as to guarantee feasibility conditions and overcome the problem of task rejection. In (Wang et al. 2015), a software-agent-based architecture provides four solutions to reconfigure the system at run-time. In addition, the agent provides three virtual processors in order to reduce the systems power consumption. However, no work deals with energy harvesting assumptions.

DAG Scheduling

Scheduling tasks modeled by DAGs has received meaningful efforts for parallelization objectives. Saifullah et al. (Saifullah et al. 2014) propose a method that eliminates inter-task dependencies so as a DAG task be transformed into a collection of independent sequential threads. The global earliest deadline first (GEDF) schedulability test is then applied to the resulting set of threads. In (Bonifaci et al. 2013), Bonifaci considers the general parameters DAG tasks synchronization regardless of their internal structures. Only two parameters related to the execution of the task model are defined: the total execution time, and the critical-path length. Recently in (Fonseca et al. 2015), the authors propose a multi-DAG model which facilitates the schedulability analysis of parallel tasks with multiple execution flows on multi-core architectures. All of these works change the nature of a DAG, and transform it into a collection of segments. Nevertheless, no one of these works consider DAGs with probabilities.

To our knowledge, no previous study has addressed the scheduling problem where tasks have to be executed with possible migration in a multiprocessor architecture with regenerative energy. As far as we know, the Probabilistic DAG model is proposed for the first time in this paper.

FORMALIZATION OF RECONFIGURABLE REAL-TIME SYSTEMS

In this section, we describe our system model that consists of a reconfigurable distributed real-time system, a harvesting energy module and an energy storage module.
Real-Time Reconfigurable System

In this paper, we are interested in dynamic task reconfiguration. An external reconfiguration event is defined as an exception or an event that leads to add/remove/update tasks. Consequently, any reconfiguration scenario may increase energy consumption and/or make some tasks to violate their deadlines. We assume that a reconfiguration scenario may affect only one processor at a given time. The system model is depicted in Figure 1.

![System Model](image)

**Figure 1: System model**

**Notation:** We consider a real-time embedded system to be formalized as Sys = \{H_w, S_w\} such that H_w is the hardware platform and S_w is the software one. H_w contains a set \( \pi \) of \( m \) processors \( \pi = \{P_1, ..., P_m\} \). We assume that preemption and migration of tasks are authorized on all processors. The multiprocessor platform is supplied by a renewable energy source and uses a set \( \beta = \{B_1, ..., B_m\} \) of \( m \) rechargeable energy storage with limited capacities. Each processor \( P_j \) in the multiprocessor platform is powered by its own storage unit denoted by \( B_j \). The software platform \( S_w \) contains a set \( \psi \) of \( N \) tasks \( \psi = \{\tau_1, ..., \tau_N\} \). Each task is assigned to a given processor according to a technique detailed afterwards.

We suppose that \( S_w(t) \) is the task set that implements the system Sys at a particular time \( t \). We denote by \( P_j \) a faulty processor (i.e. where tasks have to migrate) and \( \psi_{P_j} \) the set of tasks assigned to \( P_j \). Furthermore, in the rest of the paper a subscript "f" represents an item in a faulty processor and we denote by \( |A| \) the cardinality of the set \( A \). In our system, a task \( \tau_i \), \( i = \{1, ..., N\} \), is characterized by:

- period \( T_i \), worst case execution time (WCET) \( C_i \) in conformance with the classical task model of Liu and Layland (Liu and Layland 1973), worst case energy consumption (WCEC) \( E_{\eta_i} \) and a degree of criticality \( d_{\eta_i} \) that defines its applicative importance.

- a Probabilistic Directed Acyclic Graph (DAGP) that encodes all its possible execution traces (details are given in section IV.)

The system Sys can be reconfigured repeatedly. After any external reconfiguration scenario at a particular time \( t \) such as addition or removal of tasks, the new implementation is defined by:

\[ S_w^+(t) = S_w^-(t) \cup \xi^+/\xi^- , \]

where \( \xi^+(t) \subseteq S_w^+ \) is the set of added tasks and \( \xi^-(t) \subseteq S_w^+ \) is the set of tasks removed from Sys.

**Energy Model**

**Energy Production Model**

Let us assume that the harvesting energy is collected from different energy sources (photovoltaic, piezoelectric, thermal, ...). We suppose that the incoming power received by a given storage unit remains unchanged along time but it may be different from one storage unit to another. Let \( P_h(t) \) be the instantaneous charging rate produced by the energy source at time \( t \) and \( E(t) \) be the total energy produced over \( [0, t] \) by the power source given by the following formula:

\[ E(t) = \int_0^t P_h(t) \, dt \]

**Energy Storage Model**

In this paper, we consider a hybrid energy storage model (battery and super-capacitor). A storage unit is defined by \( B_j = (E_{j_{max}}, E_{j_{min}}) \), \( \forall j = \{1, ..., m\} \) where \( E_{j_{max}} \) and \( E_{j_{min}} \) are respectively minimal energy and maximal energy that can be stored. Note that for simplicity and without loss of generality, we assume that the energy storage can be completely depleted to as less as zero. The energy available in the storage \( B_j \) at time \( t \) is denoted by \( E_{B_j}(t) \). We also assume that each energy storage can be charged up to its capacity. Initially, every energy storage is fully-charged: \( \forall j = \{1..m\}, E_{B_j}(0) = E_{j_{max}} \).

**Energy Consumption Model**

We assume that the energy consumed by the processor \( P_j \) is equal to zero when it does not execute jobs i.e. when it is in the idle state. Furthermore, the energy consumed in the time interval \([t_1, t_2]\) is the cumulative amount of WCEC of tasks which execute on processor \( P_j \) between \( t_1 \) and \( t_2 \). The energy consumed by a job in any unit time-slot is no less than the energy produced in the same unit time-slot.

**Initial Task Assignment**

The so-called Best Fit Energy Heuristic (BFEH) is proposed for initial task assignment. This partitioning issue amounts to a Bin-Packing one which is known to be NP-hard (Michael and David 1979). Let us define \( \psi \) as a set of \( n \) periodic real-time tasks and \( \pi \) as a set of \( m \) processors. Each task in \( \psi \) is assigned to one processor of \( \pi \) as follows:
1. $\psi$ is sorted in decreasing order of the ratios given by worst case energy consumption WCEC, $E_{n_i}$ divided by deadline, $D_i$.

2. $\pi$ is sorted in decreasing order of the energy storage capacity, $E_{B_j}(t)$.

Task $\tau_i$ is assigned to processor $P_j$ if the EDF schedulability test is satisfied and the storage $B_j$ has the least residual energy. Ideally, the residual energy in the storage should be zero. We consider that all the jobs of a given task should be executed on the same processor.

**PROBABILISTIC TASK DAG MODEL**

In this section, we introduce a new task model called probabilistic directed acyclic graph (DAGP) attached to each task in the software platform $S_w$.

**Motivation**

Typically, the code of any task is constructed from one or more control structures such as the "if-then-else" statements. Two jobs $\tau_{i,k}$ and $\tau_{i,h}$ of task $\tau_i$ may execute different parts of the code. Hence, an "execution flow" is defined as the path used by a job throughout its execution. In real world, the various possible execution flows do not have the same chance to be used. A probability is attached to execution flow so as to express the chance or the risk that a job performs this execution flow.

**Formalization**

A DAGP model is attached to every task $\tau_i, i = \{1..N\}$ as depicted in Figure 2. This model is a graph $G_i = (V_i, E_i)$ where $V_i = \{\tau_{i,1}, ..., \tau_{i,n_i}\}$ is the set of task nodes that represent the sub-tasks of $\tau_i$. $n_i$ is the number of sub-tasks in $G_i$, and $E_i$ is the set of directed edges that represent dependencies between nodes in the graph $G_i$. Each edge is labeled by constant $p$ which designates the inter-arrival period. The inter-arrival time $p$ between $\tau_{i,k}$ and $\tau_{i,h}$ is defined as the amount of time that must elapse after the execution of $\tau_{i,k}$ and before the task $\tau_{i,h}$ can be triggered. We suppose that $p$ is the same among all subtasks. We assume that DAGP $G_i$ attached to task $\tau_i, i = \{1..N\}$ is characterized by a set $F_i = \{F_{i,1}, ..., F_{i,m_i}\}$ which denotes the set of all possible execution flows of $G_i$. $m_i$ is the number of execution flows in $G_i$. Each execution flow $F_{i,j} = \{P_{i,j}, V_{i,j}, E_{i,j}\}$ is characterized by a probability $P_{\tau_{i,j}}$, a set of nodes $V_{i,j}$ and a set of edges $E_{i,j}$. We associate a $dc_i$ attribute to each task, that defines its execution emergency. Task $\tau_i, i \in \{1..N\}$ is characterized by sextuplet $(G_i, C_i, D_i, T_i, E_{n_i}, dc_i)$, where i) $G_i$ the DAGP associated to task $\tau_i$, ii) $C_i$ the maximum number of CPU clock cycles needed to complete a job instance of the task $\tau_i$ called WCET iii) the relative deadline of the task, $D_i$, iv) $T_i$ is the period of the task. We assume that $D_i$ is equal to $T_i$, v) $E_{n_i}$ is the energy requirement of the task $\tau_i$, the amount of energy required for its execution, called WCEC and vi) $dc_i$ the execution emergency level of the task $\tau_i$. We introduce in the following the system model terminology.

**Definition 1.** The utilization factor of task $\tau_i$ is denoted by $U_i$ and defined as follows:

$$U_i = \frac{C_i}{T_i}$$

The CPU utilization of processor $P_j, j = \{1..m\}$ is denoted by:

$$U_{P_j} = \sum_{i=1}^{n} U_i$$

where $n$ is the number of tasks assigned to processor $P_j$.

**Definition 2.** The set of $n$ tasks assigned to processor $P_j$ is schedulable under a scheduling policy if the processor utilisation factor $U$ is no greater than the schedulable utilisation $U_x$ as denoted by:

$$\sum_{i=1}^{n} C_i \leq U_x$$

where:

$$U_x = \left\{ \begin{array}{ll} n(2^{\frac{1}{n}} - 1) & \text{if RM is applied} \\ 1 & \text{if EDF is applied} \end{array} \right.$$  

**Definition 3.** The amount of energy consumed by processor $P_j, j = \{1..m\}$ is defined as the sum of WCEC $E_{n_i}$ of all $n$ tasks assigned to $P_j$:

$$E_{P_j} = \sum_{i=1}^{n} E_{n_i}$$

**Definition 4.** The WCET of an execution flow $F_{i,j}$ of task $\tau_i, i = \{1..N\}$ is defined as the cumulative amount of WCET of all nodes of the set $V_{i,j}$ plus the inter-arrival period $p$:

$$C_{F_{i,j}} = \sum_{\tau_{i,k} \in V_{i,j}} C_{\tau_{i,k}} + p \times (|V_{i,j}| - 1)$$
The critical execution flow $F_{i,j}$ of task $\tau_i$ is defined as the execution flow with the longest execution time. **Definition 5.** The WCET of task $\tau_i$ is defined as the length of the critical path as follows:

$$C_i = \max (C_{F_{i,j}}), j \in \{1..m_i\}$$ (6)

In the context of the DAGP model, we introduce the notion of probabilistic utilisation factor $U_{Pr_{\tau_i,k}}$ which denotes the utilisation factor of sub-task $\tau_{i,k}$ with probability $P_r$.

**Definition 6.** According to Shin and Choi (Shin and Choi 1999) the utilisation factor of sub-task $\tau_{i,k}$ is denoted by:

$$U_{Pr_{\tau_i,k}} = P_r \times \frac{C_{\tau_{i,k}}}{T_{\tau_{i,k}}}$$ (7)

**Definition 7.** The workload $U_{F_{i,j}}$ of execution flow $F_{i,j}$ of task $\tau_i$ is defined as the cumulative amount of utilization for all sub-tasks in $V_{i,j}$:

$$U_{F_{i,j}} = \sum U_{Pr_{\tau_{i,k}}} \tau_{i,k} \in V_{i,j}$$ (8)

**Definition 8.** The energy, $E_{F_{i,j}}$, consumed during the execution flow $F_{i,j}$ is denoted by:

$$E_{F_{i,j}} = K \times U_{F_{i,j}}, K = C \times V^2 \times F$$ (9)

where $C$ is a constant that depends on the processor identity. We denote respectively by $F$ and $V$ the frequency and voltage of the system.

**Definition 9.** The worst case energy consumption $E_{n_i}$ of a task $\tau_i$ is defined as follows:

$$E_{n_i} = Max (E_{F_{i,j}}), j \in \{1..m_i\}$$ (10)

**CASE STUDY**

Let us consider a real-time embedded system Sys such that $H_r = \{P_1, P_2, P_3\}$ where each processor supports only one operating frequency. We assume that task preemption and migration are authorized on all processors. The harvested energy is stored in a set of three batteries / super-capacitors $\beta = \{B_1, B_2, B_3\}$ where $P_1$ is supplied from $B_1 = 45$, $P_2$ supplied from $B_2 = 100$ and $P_3$ supplied from $B_3 = 40$. The software platform initially contains a set of three periodic tasks $\tau_1, \tau_2, \tau_3$ depicted in Table 1. In this case study, the EDF policy is applied for the three processors.

Initially, the task set is assigned to the multiprocessor platform according to the Best Fit energy heuristic (BF EH) presented previously. $\tau_1, \tau_2$ and $\tau_3$ are assigned to processor $P_1, P_2$ and $P_3$ respectively. Due to the Cheddar implementation, the feasible scheduling result of the system Sys is shown in Figure 3. The system is feasible since the CPU loads of the multiprocessor platform are $U_{P_1} = 0.55$, $U_{P_2} = 0.56$ and $U_{P_3} = 0.4$. The energy consumptions are $E_{P_1} = 7$, $E_{P_2} = 13$ and $E_{P_3} = 5$ energy units.

**Table 1: Initial System Configuration**

<table>
<thead>
<tr>
<th>Task</th>
<th>$C_i$</th>
<th>$T_i$</th>
<th>$D_i$</th>
<th>$E_{n_i}$</th>
<th>$d_{e_i}$</th>
<th>(m,k)-firm</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau_1$</td>
<td>11</td>
<td>20</td>
<td>20</td>
<td>7</td>
<td>A</td>
<td>(1,2)</td>
</tr>
<tr>
<td>$\tau_2$</td>
<td>9</td>
<td>16</td>
<td>16</td>
<td>13</td>
<td>B</td>
<td>(1,2)</td>
</tr>
<tr>
<td>$\tau_3$</td>
<td>10</td>
<td>25</td>
<td>25</td>
<td>5</td>
<td>C</td>
<td>(1,2)</td>
</tr>
</tbody>
</table>

**Figure 3:** Initial schedule for the system

**Reconfiguration scenario1:** Suppose that at time $t_1$, an external reconfiguration scenario is applied to add a new task $\tau_4$ (Table 2) to processor $P_1$. After addition, as shown in Figure 4, the schedule is infeasible since $\tau_4$ misses its deadline equal to 20. The energy consumed by processor $P_1$ increases up to 20 energy units.

**Figure 4:** Schedule after addition of task $\tau_4$

**Reconfiguration scenario2:** Suppose that at time $t_2$ after $t_1$, a second external reconfiguration scenario is applied to add task $\tau_5$ (Table 2) to processor $P_2$. After addition to $P_2$ as shown in Figure 5, the CPU load is equal to 1.0625. The energy consumption, $E_{P_2}$, also increases on the hyper-period $H$ ($H=[\text{lcm}(T_4, T_5)] = 48$) and becomes equal to 95 units of energy. The schedule is infeasible since task $\tau_5$ misses its deadline at 48 and completes at $t = 51$.

**Figure 5:** Schedule after addition of task $\tau_5$

**Reconfiguration scenario3:** Suppose that at time $t_3$ after $t_2$ an external reconfiguration scenario is applied to add task $\tau_6$ (Table 2) to processor $P_1$. After addition, as shown in Figure 12, the CPU load is equal to 1.65.
Figure 6: DAGP $G_1$ associated to task $\tau_1$

Figure 7: DAGP $G_2$ associated to task $\tau_2$

Figure 8: DAGP $G_3$ associated to task $\tau_3$

Figure 9: DAGP $G_4$ associated to task $\tau_4$

Figure 10: DAGP $G_5$ associated to task $\tau_5$

Figure 11: DAGP $G_6$ associated to task $\tau_6$

Figure 12: Schedule after addition of task $\tau_6$

Figure 13: Scheduling of the system after the addition of the task $\tau_7$

Figure 14: Energy consumption of processor $P_3$

**Reconfiguration scenario4:** Suppose that at time $t_4$ after $t_3$ an external reconfiguration scenario is applied to add task $\tau_7$ (Table 2) to processor $P_3$. The CPU utilisation of processor $P_3$ becomes $U_{\tau_7} = 0.8$, hence the timing constraint is satisfied but as shown in Figure 13, the schedule is infeasible since the energy constraint is violated. As the energy consumption of $E_3$ is equal to 80 on the hyper-period $H (H = \lcm(T_3, T_7) = 75)$, $\tau_3$ stops at $t = 20$ before completing execution since there is no sufficient energy in the storage unit. The energy consumption profile of processor $P_3$ is presented in Figure 14.

<p>| Table 2: System Reconfiguration Scenarios |</p>
<table>
<thead>
<tr>
<th>Task</th>
<th>$C_i$</th>
<th>$T_i$</th>
<th>$D_i$</th>
<th>$En_i$</th>
<th>$d_{c_i}$</th>
<th>$(m,k)$-firm</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\tau_4$</td>
<td>10</td>
<td>20</td>
<td>20</td>
<td>13</td>
<td>D</td>
<td>(1,2)</td>
</tr>
<tr>
<td>$\tau_5$</td>
<td>6</td>
<td>12</td>
<td>12</td>
<td>14</td>
<td>D</td>
<td>(1,2)</td>
</tr>
<tr>
<td>$\tau_6$</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>A</td>
<td>(1,2)</td>
<td></td>
</tr>
<tr>
<td>$\tau_7$</td>
<td>6</td>
<td>15</td>
<td>15</td>
<td>13</td>
<td>E</td>
<td>(1,2)</td>
</tr>
</tbody>
</table>
The reconfiguration scenario increases the energy consumption and/or pushes some tasks to violate the corresponding deadline. In order to re-establish the system feasibility, we propose four solutions which are detailed in section VI.

NEW SOLUTIONS FOR FEASIBLE LOW-Power REAL-TTME RECONFIGURABLE SYSTEMS

The problem of scheduling probabilistic directed acyclic graphs (DAGP) on a reconfigurable multiprocessor platform under hard real-time and energy constraints is known to be NP-complete (Michael and David 1979). We assume that all sources such as hardware failures may occur and impose a dynamic software reconfiguration for maintaining feasibility. This paper addresses the even more difficult problem of scheduling on reconfigurable systems that entirely rely on energy harvesting with limited capacity storage. We provide four approaches to address these challenges.

Motivation

A run-time external reconfiguration scenario is a dynamic operation allowing the addition/removal of the assumed DAGP tasks. Thereafter, some tasks may miss their hard deadlines and the energy constraint may be violated. Hence, checking the system feasibility after any reconfiguration scenario is of utmost importance. Further, each processor in the multiprocessor platform should satisfy the hard real-time constraint in which the CPU processor utilization is no greater than the schedulable utilization \( U_x \). In addition to the energy constraint where the processor’s energy consumption is at most equal to the energy harvested in the storage associated to the processor. If the system feasibility is satisfied then the system operates normally. Besides, if at least one of the feasibility conditions is violated then the dynamic software reconfiguration solutions are applied one by one in order to reconfigure the system and re-establish system feasibility. The four solutions are performed in a hierarchical order:

- Modify the inter-arrival period of DAGP tasks so as to decrease the global load,
- Decompose each DAGP task of the faulty processor to a set of hipsters and let them migrate to other non-faulty processors,
- Degrade the quality of service on each faulty processor. Tasks may be executed according to (m,k)-firm constraints,
- Delete hipsters or DAGP tasks so as to minimize the global deadline miss ratio.

Formalization

In this section, we formalize the solutions.

Solution 1: Inter-Arrival Period Modification (IAPM)

It consists in modifying the inter-arrival period for two consecutive sub-tasks so as to decrease the global load.

**Proposition 1:** To re-obtain system feasibility on the faulty processor, the new inter-arrival period \( p \) should be equal to:

\[
U_x - \sum^n_{i=1} \sum_{j=1}^{[V_{i,c}]} \frac{C_{i,j}}{T_i} \geq \sum^n_{i=1} \sum_{j=1}^{[V_{i,c}]} \frac{([V_{i,c}] - 1) * p}{T_i}
\]

**Proof:** According to formula (3), a given task set is assumed to be schedulable under a scheduling policy if the utilisation processor factor \( U_x \) is no greater than the schedulable utilisation \( U_x \):

\[
\sum^n_{i=1} \frac{C_i}{T_i} \leq U_x
\]

According to definition 4 the WCET \( C_i \) of task \( \tau_i \), \( i \in \{ 1..N \} \), is equal to \( C_{i,j} + ([V_{i,c}] - 1) * p \) by replacing \( C_i \) by its value in formula (3) the feasibility condition can be calculated as follows:

\[
\sum^n_{i=1} \sum_{j=1}^{[V_{i,c}]} \frac{C_{i,j} + ([V_{i,c}] - 1) * p}{T_i} \leq U_x
\]

Then, the inter-arrival period \( p \)

\[
p \leq \frac{U_x - \sum^n_{i=1} \sum_{j=1}^{[V_{i,c}]} \frac{C_{i,j}}{T_i}}{\sum^n_{i=1} \sum_{j=1}^{[V_{i,c}]} \frac{([V_{i,c}] - 1) * p}{T_i}}
\]

**Running-example 1:** Let us consider the reconfiguration scenario1, the solution1 "Inter-arrival period modification" is applied to processor \( P_1 \). According to formula (11), the inter-arrival period \( p \) is decreased up to 1. Hence the WCET of tasks \( \tau_4 \) and \( \tau_1 \) are decreased to 8 and 9. Therefore, the processor utilization factor is modified to be \( U_{p_1} = 0.85 \) from \( U_{p_1} = 1.05 \). As shown in Figure 15, the schedule on \( P_1 \) is feasible since \( U_{p_1} = 0.85 \).

Figure 15: Schedule after Inter-arrival period modification

Solution 2: Decomposition and Migration of Branches

This solution is divided into two steps:

- Decompose each DAGP task of the faulty processor to a set of hipsters and let them migrate to other non-faulty processors,
- Degrade the quality of service on each faulty processor. Tasks may be executed according to (m,k)-firm constraints,
• First step: selection of a branch or group of branches and make them migrating to other non-faulty processors in order to re-establish the system feasibility,

• Second step: selection of processors into which the migrant branches will be assigned.

Step1: Hipster Selection Heuristic

Proposition 2: The task with the lowest degree of criticality will be decomposed into a set of hipsters. Then the current critical execution flow will be removed from the DAGP.

Step2: Processor Selection Heuristic

Proposition 3: We sort the set of processors for which the candidate hipsters can be affected. Then, sort this set in increasing order of energy availability in storage unit.

Running-example 2: When the second reconfiguration scenario is applied, Solution 2 "Decomposition-Migration" is as follows. Step1: Sort the task set \( \{\tau_2 = (9,16,16,13), \tau_5 = (6,12,12,14)\} \) in a non-decreasing order of critical level then the task \( \tau_2 \) is selected to be decomposed. Let \( G_2 = (F_2, V_2, E_2) \) be the DAGP associated to task \( \tau_2 \) as depicted in Figure 7.

Let, \( F_2 = (F_{2,1}, F_{2,2}, F_{2,3}, F_{2,4}) \) be the set of all execution flows of \( \tau_2 \) such that \( F_{2,1} \) is the critical execution flow. The new DAGP \( G_2 \) associated to task \( \tau_2 \) after elimination of the critical execution flow \( F_{2,1} \) is shown in Figure 16 and new parameters of \( \tau_2 \) become equal to \((7,16,16,12)\). Step2: By migrating the hipster \( F_{2,1} \) to processor \( P_3 \), the system feasibility is re-established as presented in Figure 17. The schedule is feasible since \( U_{p_0} = 0.96 \) and \( U_{p_2} = 0.93 \). The total energy consumptions of processor \( P_2 \) and \( P_3 \) are \( E_{p_2} = 92, E_{p_3} = 25 \).

![Figure 16: DAGP \( G_2 \) associated to task \( \tau_2 \) after elimination of the critical execution flow \( F_{2,1} \)](image)

Running-example 3: When the third reconfiguration scenario is applied, Solution 3 "Degradation" is as follows.

\[
\sum_{i=1}^{n} \frac{C_i}{T_i + k_i} \leq U_e, \quad \theta = \frac{U_e}{\sum_{i=1}^{n} \frac{C_i}{T_i}}, \quad \gamma = \frac{E_{p_i}}{T_i}
\]

Therefore, we accept that in the faulty processor \( P_1 \) the tasks be executed under \((1,2)\)-firm constraints which indicate that the deadlines of at least 1 instance among 2 consecutive ones must be met for tasks \( \tau_1 \), \( \tau_4 \) and \( \tau_6 \).

Solution 4: Removal Hipster (RHH)

Delete hipsters or DAGP tasks so as to minimize the global deadline miss ratio. For any faulty processor \( P_f \), we associate to each task in \( \psi_{pf} \) a density denoted by:

\[
\gamma = \frac{E_{p_i}}{T_i}
\]

Proposition 5: We sort all tasks in increasing order of densities so that we can reject one by one those with highest densities until the remaining utilization factor of the faulty processor is lower than \( U_e \) and the total energy consumption is below \( E_{Bf(1)} \).

Running-example 4: When the reconfiguration scenario 4 is applied, Solution 4 "Removal Hipster" is performed to processor \( P_3 \). According to formula (13) \( \gamma_3 = \frac{1}{3} \) and \( \gamma_7 = \frac{1}{5} \). Then, task \( \tau_7 \) has the highest density. By removing the critical execution flow \( F_{7,1} \), parameters of \( \tau_7 \) become \((5,15,15,4)\). Therefore, the processor utilization factor is modified to be \( U_{p_3} = 0.73 \) instead of \( U_{p_5} = 0.8 \) and the total energy consumption \( E_{p_3} = 35(E_{p3}(t) = 40 \). In this case study, the solution "Removal Hipster" reduces the total energy consumption of processor \( P_3 \) with up to 56.25\%.
EXPERIMENTAL STUDY

According to the solutions formulated above, Algorithm1 evaluates a set of unpredictable reconfiguration scenarios applied repeatedly during the running execution of the system and provides four solutions in order to re-establish the system feasibility.

Algorithm 1 Main algorithm
while 1 do
    Reconfig();
    if feasible(Pj) = false then
        Inter-arrival period Modification(Pj, \psi_{Pj})
        if feasible(Pj) = false then
            Decomposition Migration(Pj, \psi_{Pj}, B_f(t), \pi, \psi)
            if feasible(Pj) = false then
                Degradation(Pj, \psi_{Pj})
                if feasible(Pj) = false then
                    Removal Hipster(\psi_{Pj}, B_f(t))
            end if
        end if
    end if
end while

Decomposition-Migration DAGP Solution

Hipster Selection Heuristic
Algorithm 2 depicts the pseudo-code of hipster selection heuristic (HSN). For a faulty processor Pj, Branch Selection Heuristic sorts tasks in increasing order based on criticality level dc, then, the task \tau_{acc} with the lowest dc is selected to be decomposed. We calculate the WCET and WCEC of each execution flow in \tau_{acc}. In order to get a feasible execution in the faulty processor Pj, the workload should be no greater than U_x and the total energy consumption should be no greater than the amount of energy available in the storage B_f. Therefore, the scheme deletes the current critical execution flow F_{acc}, and recalculates the new one until the feasibility conditions are satisfied. The overall run-time computational complexity of Hipster Selection Heuristic is O(N^2).

Processor Selection and Assignment Heuristic
Algorithm 3 depicts the pseudo-code of processor selection and assignment heuristic (HSAH). The overall run-time computational complexity of Processor Selection and Assignment Heuristic is O(m).

Removal Hipster Heuristic
The pseudo-code of the Removal Hipster Heuristic (RHH) is given in algorithm 4. For a faulty processor Pj we associate to each assigned task a density defined as the ratio of WCET over the period T_i. We sort all tasks in an increasing order of densities so as to reject critical execution flows with highest densities, one by

Algorithm 2 Hipster selection heuristic
Input: \psi_{pf} = \{\tau_1, ..., \tau_n\} tasks’ set assigned to the faulty processor P_f; F_{acc} = \{F_{acc,1}, ..., F_{acc,N}\} denotes the set F_{acc} of all possible execution flow of G_{acc}; The energy available in the storage of processor P_f at a time t B_f(t).
Output: Candidate branches.
Sort tasks’ set \psi_{pf} in increasing order of criticality level
\tau_{acc} \leftarrow min(\psi_p)
for k = 1 to N do
    Calculate WCET(F_{acc,k}), Calculate E_{F_{acc,k}}
end for
repeat
    F \leftarrow Max(F_{acc})
    Delete(F, G_{acc})
    C_{\tau_{acc}} \leftarrow Max(C_{F_{acc,i}}), E_{\tau_{acc}} \leftarrow Max(P_{F_{acc,i}})
    Calculate U_{pf}
    Calculate E_{pf}
until U_{pf} < U_x and E_{pf} < B_f(t)

Algorithm 3 Processor Selection and Assignment Heuristic
Input:Processors’ set \pi = \{P_1, ..., P_m\};
Output: Branch assigned to non faulty processor.
P \leftarrow \pi / \{P_f\}
for i = 1 to m - 1 do
    if U_i \leftarrow U_i + \frac{C_{\tau_{acc}}}{T_{\tau_{acc}}} < U_x then
        insert(P_i, \pi_{acc})
    end if
end for
Affect F_{acc} to the candidate processor which has the highest energy level.
one, until the remaining utilization of the faulty processor is lower than \( U_z \), and the total energy consumption is below \( E_{B_f(t)} \). The overall run-time computational complexity of Removal Hipsters Heuristic is \( O(n) \).

Algorithm 4 Removal Hipster

Input: \( \psi_{B_f} = \{\tau_1, ..., \tau_n\} \) task set assigned to faulty processor \( P_f \); The energy available in the storage of processor \( P_f \) at time \( t \); \( B_f(t) \).

Output: feasible tasks set

for \( i = 1 \) to \( n \) do

\[ \gamma_i = \frac{E_{B_f}}{T_i} \]

end for

sort task set \( \psi_{B_f} \) in decreasing order of task densities

\[ T_{acq} \leftarrow T \]

for \( i = 1 \) to \( n \) do

if \( U_{B_f} \geq U_z \) and \( E_{B_f}(t) \geq E_{B_f}(t) \) then

reject critical execution flow \( F_c \) of \( n \)th task

done with task rejection, break

end if

end for

solutions. Figure 19 presents the percentage of energy gain. It can be derived from this figure that the solution RHH achieves energy savings of up to 67\% when compared to the energy consumption of the initial task schedule. However, the solution DMH provides energy savings up to 5\%.

CONCLUSION AND DISCUSSION

We firmly believe that nowadays it has become crucial to investigate new models and techniques to schedule contemporary applications subject to energy and real-time requirements, especially when unpredictable reconfiguration scenarios occur at run time. To the best of our knowledge, the work reported in this paper is the first one that focuses on scheduling of real-time periodic tasks in a reconfigurable multiprocessor energy harvesting real-time system. In this paper, we have presented a partitioning heuristic in the energy harvesting context, namely 3F-EH, in order to build the initial system configuration. However, when an external reconfiguration scenario is applied, the system may evolve towards an infeasible state. We have proposed a new task model where a Probabilistic Directed Acyclic Graph DAGP is attached to each task. Indeed, the probability of execution flows may reduce the percentage of missed dead-

![Figure 18: Percentage of satisfied deadlines, when a set of reconfiguration scenario is applied repeatedly during the execution time](image1)

![Figure 19: Percentage of energy gain when a set of reconfiguration scenario is applied repeatedly during the execution time](image2)
lines and increases the energy availability in the storage unit. We have proposed four approaches to re-establish the system feasibility. Extensive simulation experiments show that the proposed four solutions achieve energy savings up to 67%, and reduce the deadline miss ratio up to 10%.

REFERENCES


APPLICATION OF INTELLIGENT CONTROL ALGORITHMS FOR THERMAL COMFORT AND ENERGY SAVING IN THE COMPUTER LABORATORY

Mariusz Nowak
Andrzej Urbaniak

Institute of Computing Science
Poznan University of Technology
Piotrowo 2, 60-965 Poznan, Poland
E-mail: {Mariusz.Nowak|Andrzej.Urbaniak}@put.poznan.pl

KEYWORDS
Intelligent control, object modelling, microclimate comfort, energy saving, HVAC.

ABSTRACT
In the paper there are presented the problems of optimal control for indoor thermal comfort in laboratory room equipped with HVAC (Heating, Ventilation and Air Conditioning) systems. It is very important that the design of such HVAC control systems allows to minimize the energy consumption and to maximize microclimate comfort sensation. There are discussed the results of modeling the laboratory room using building construction data. The mathematical model of the laboratory room has been verified using meteorological data from the weather station, taking into account the conditions of HVAC actuators. Intelligent control algorithms are used in the structure of climate comfort control system. In the paper the commercial HVAC control system and intelligent HVAC control system are compared.

INTRODUCTION
According to the European Commission, traditional buildings consume about 40% of the total fossil fuel energy in European Union (EU), half of this energy is spent for indoor climate conditioning. The building sector contributes up to 30% of global annual greenhouse gas emissions. Increasing energy efficiency of buildings is one of the fundamental issues they are working on the authorities and scientific institutions of the EU. In March 2007 the EU’s leaders endorsed an integrated approach to climate and energy policy that aims to combat climate change and increase the EU’s energy security while strengthening its competitiveness. They set 20/20/20 targets to be meet by year 2020: 20% reduction of GHG emissions by 2020 in comparison with to 1990, 20% share of renewable energy in final energy consumption, 20% reduction in EU primary energy consumption, in comparison with projected levels, to be achieved by improving energy efficiency (see On the Energy Performance of Buildings).

In the rooms are emitted pollution, heat and moisture gains. To maintain an appropriate level of microclimate comfort in a closed rooms must ensure adequate air exchange, to lead the process of heating or cooling the air inside the rooms. It is assumed, that the realization of processes of heat and mass transfer associated with costs that are the most important component of the overall costs of the buildings. Continuing increases in energy prices and new European Union guidelines on minimizing the consumption energy, forces us to seek ways to reduce the energy consumption of air conditioning systems. These systems must also provide adequate microclimate comfort. Therefore, it is necessary to develop advanced control systems to minimize the energy consumption of air conditioning systems for buildings. The desired thermal comfort in buildings depends on the power efficient systems.

Thus, this paper presents a proposal to reduce energy consumption while ensuring a feeling of microclimate comfort by users, through the use of advanced computer control of air conditioning system. The research has been performed in the laboratory room in Lecture and Congress Centre of the University’s Technical Library Building.

This paper presents results of a comparison of the commercial HVAC control system, that was installed in laboratory room and intelligent HVAC control system, which was proposed by the authors of the paper. Synthesis of intelligent control algorithms have been made in the universal computing environment, scientific and technical - Matlab & Simulink. Simulation studies were implemented in Matlab & Simulink environment. As mentioned above, the crucial part of energy consumption in a building is connected with heating and cooling devices.

For this reason, the first step would be the preparation of mathematical model of a laboratory room. To create a mathematical model of the laboratory room we used meteorological data were obtained from the weather station localized near the building in Poznan. Then, the next step presents the realisation of verification a mathematical model of analysed laboratory room and realisation of a thermal comfort model of this laboratory room. The proposed thermal comfort model can be used for the study of HVAC systems. The model is also useful for the studies of control strategies as well as for finding the solutions for reducing the electrical energy consumption.
and for maintaining acceptable indoor air conditions related to thermal comfort. Thus, in the paper there are suggested control strategies for reducing energy consumption in relation to commercial HVAC control system and maintaining these acceptable indoor air conditions related to thermal comfort.

MATHMATIC MODEL OF COMPUTER LABORATORY

The authors of the article have developed a mathematical model of the computer laboratory. To create the model, a building construction data and information about installed devices provide indoor microclimate comfort were necessary. Selected laboratory room that are serviced by one air handing unit, one chilled water generator and one heating system installation. The dimensions of the laboratory room are as follows: length – 10 [m], width – 5 [m], height – 4 [m]. Room is equipped with 2 doors and skylight ceiling and also with two radiators, two fan-coil units, two circular air diffusers and two exhaust air valve.

Building modeling taking into account the all aspects of the physical is very difficult. In order to formulate the reduced model of the building, usual simplifying assumptions as time invariant parameters, uniformly distributed properties, etc. are considered. Models derived from physical relations are naturally represented in state-space by a set of first order differential equations (Dounis and Ciaiosos 2009; Hazyuk et al. 2012a; Fanger 1982). The choice of the object model is determined by the type of used control algorithm. On the market there are dedicated software tools to simulate building thermal behavior, such as CODYBA, TRNSYS, Comfie, ESP-R, EnergyPlus and Simbad (Nowak and Urbiak 2005; Nowak and Urbiak 2007; Nowak and Urbiak 2010a; Hazyuk et al. 2012a).

The low-order model estimated in this paper is implemented in IBPT (see International Building Physics Toolbox) toolbox in Matlab Simulink. The IBPT toolbox is an open source and is available for the researchers, students and practitioners in the building physics area for free downloading. Low-order building models used for control purpose are most often derived from linear network representations with lumped parameters (Hazyuk et al. 2012a). Because the laboratory room is surrounded by other rooms, for mathematical modeling, a simple one-node RC (2R-C) model of partition has been used. General scheme is shown in Figure 1. \( T_0 \) and \( T_i \) – respectively, the temperature outside and inside the room, \( R_{out} \) and \( R_{in} \) – respectively, thermal resistance of element, \( C \) – heat capacity of element, \( q \) – additional heat source, \( T \) – modeled temperature.

![Figure 1: One-node RC model of partition](image)

In approach used the principle of analogy between two different physical domains that can be described by the same mathematical equations. The linear electrical circuit represents the building and the state-space equations are obtained by solving that circuit. The temperature is equivalent to voltage, the heat flux to current, the heat transmission resistance is represented by electrical resistance and the thermal capacity by electrical capacity. The equivalent circuit of the building is obtained by assembling models of the walls, windows and internal mass. The roof and the floor also can be seen as some kind of wall, and therefore they are modeled by the same network structure. The internal thermal mass is usually represented by a single capacity. Windows do not accumulate thermal energy and therefore they are represented as simple resistances. Ventilation and infiltrations are also modeled by a resistance (Hazyuk et al. 2012a). During the modeling of temperature changes inside the room is assumed that the parameters of the partitions are constant over time. The fundamental equation is as follows (1):

\[
CT = \frac{T_0 - T}{R_{out}} + \frac{T_i - T}{R_{in}} + q
\]  

(1)

The internal heat flux is the sum of all the internal free gains (from building occupants, electrical appliances, solar radiation through windows) and contributions from radiators. The output of the model is the indoor temperature. This temperature is influenced by four different inputs: outdoor air and ground temperatures, solar radiation and internal sources. Outdoor temperature and solar radiation are uncontrollable sources but they can be measured – they are measurable disturbances. For the operating temperature range of the building, the model is considered to be linear. Information about climate parameters outside the building from the meteorological station was necessary to create a mathematical model of the laboratory room. From the meteorological station the following data has been obtained: external temperature, illumination, wind speed, that was necessary to create a mathematical model of laboratory room.

The meteorological data from the weather station were used to identifying the parameters of the real building. The parameters of the mathematical model have been identified from input-output data records obtained by simulating a detailed model of laboratory room. Verification of mathematical model had been made for September 2014. During that period there has been no people in the room and HVAC devices in that room were turned off.

The course of internal temperatures changes for the modeled room, simulated in Matlab Simulink, as well as for the real room was identical.

THERMAL COMFORT MODEL

It is believed that thermal comfort is a non-linear result of the interaction between environmental and personal factors as well as subjective feelings. The most important environmental factors are: air temperature, mean radiant temperature, air velocity and relative humidity. Personal
factors include activity level and clothing insulation. All issues related to thermal comfort are widely discussed in the literature (Nowak and Urbaniak 2010b; Nowak and Urbaniak 2011a; Ghafranman et al. 2014; Gomez-Otero et al. 2012; Jazizadeh et al. 2014). To evaluate human thermal comfort we used indirect indicators that combines the environment parameters and feelings of comfort. Parameters of environments and indirect indicators: operating temperature, effective temperature were used in the simulation experimental. The PMV index (Predictive Mean Vote), PPD index (Predicted Percentage Dissatisfied) and DR index (Draught Rate) were used in the simulation. The precise characteristics of indicators has been included in the previous papers (Nowak and Urbaniak 2005; 2007, 2010a, 2010b, 2011a, 2011b, 2013; Fanger 1982).

INTELLIGENT ALGORITHMS OF THERMAL COMFORT CONTROL

The main task of the commercial HVAC control system installed in the laboratory room is to provide: the required room temperature and required fresh air in the ventilation system. The additional task of the commercial HVAC control system is to minimize the energy consumption. These goals are accomplished through the use of bang-bang controllers and PID controllers. However, energy consumption optimization is based on the use of method, where settings of the air parameters use the occupancy schedule. This is classical reaction control system. In the commercial HVAC control system, the PID controllers are rarely tuned. Change of control object parameters in building automation system are not included. Therefore other, intelligent control strategies are being considered. The use of intelligent methods to the control systems of buildings started in the 90s of the last century. Artificial Intelligence (AI) techniques were applied to the control of conventional buildings. At the beginning the intelligent controllers, optimized by the use of evolutionary algorithms were developed for the control of the subsystems of an intelligent building (Dounis and Caraiscos 2009). In several works, a lot of proposals to modify the classical control strategy, by the use of artificial intelligence can be found. The use of fuzzy logic, neural networks, genetic algorithms for building thermal control has been investigated too (Dounis and Caraiscos 2009).

The use of Model Predictive Control (MPC) for Building Automation System is proposed in this paper. The MPC approach has several features that make it suitable for the problem encountered in intermittently heated buildings. MPC is able to use the occupancy schedule and weather forecasts for optimal temperature control and MPC optimizes not only the comfort but also an energy criterion. The MPC may be used to makes a tradeoff between energy savings and thermal comfort. Therefore, this approach has the advantage over classical control. MPC is able to handle implicitly the constraints in Multi-Input, Multi-Output (MIMO) systems. MPC gives the best results as compared to other control algorithms, which was presented in the several papers and books (Tatjewski 2002; Mirinejad et al. 2012; Oldewurtel et al. 2012; Hazik et al. 2012b). Thus, the authors of the paper, have decided to check the suitability MPC algorithm to control the air temperature and air ventilation of the air in the laboratory room in building.

The analysis of basic implementation and modification of predictive controllers in air-conditioner system were presented in previous papers (Nowak and Urbaniak 2011a; Nowak and Urbaniak 2011b; Nowak and Urbaniak 2013). MPC algorithms are defined by the process model related to control purposes. The design of the control system is characterized by four main steps: process modelling, cost function definition, cost function optimization and redefine strategy horizon (Nowak and Urbaniak 2011; Nowak and Urbaniak 2013; Tatjewski 2002). The control rule is given by the following general optimization problem (2):  

\[
\min J(i) = \sum_{i=1}^{N} (x(i) - y(i))^2 + \lambda \sum_{i=0}^{N-1} \Delta u(i)^2
\]

(2)

where: \(x(i)\) - i-th reference, \(y(i)\) - i-th measured output, \(\Delta u(i)\) - i-th manipulated variable change, \(\lambda\) - weighting coefficient penalizing relative big changes in \(\Delta u, Ny\) - prediction horizon, \( Nu\) - control horizon.

Setting the set points for air conditioning control system is realized by minimizing the objective function. In general, the objective function consists of the cost of non-compliance with the parameters of thermal comfort in the laboratory room, cost of realization of air changes and cost of air conveying. MPC usually optimizes the cost function from equation (2), other formulations of economic and discomfort criteria can also be employed. MPC algorithm in the DMC (Dynamic Matrix Control) version has been used in the experimental studies (Tatjewski 2002).

SIMULATION RESULTS

The mathematical model of thermal comfort and mathematical model of laboratory room have been implemented in Matlab & Simulink along with the use of intelligent algorithms in simulation investigations. The main task of the proposed intelligent control system is to provide thermal comfort and minimize energy consumption (Nowak and Urbaniak 2010; Siroky et al. 2011; Hazik et al. 2012b; Aflah et al. 2013; Harish and Kumar 2014). Simulation studies have been conducted by taking into account two conflicting criteria: optimization of the PMV index value, and energy saving. The problem of energy saving has been analyzed by assuming that the PMV index value must be included in the limit: -0.5 < PMV < +0.5. Simulation have been performed in September 2014, taking into account the meteorological data for this period. Setpoint values of indoor air parameters: temperature – 21 °C and humidity – 50 [%]. The result of the simulation of changes of thermal parameters in laboratory room were compared with the real data which were obtained from the BMS.

Results of research of energy consumption (power consumed by the radiator, cooler with fan and humidifier) for a system with predictive control algorithms with model of laboratory room and the system using only the traditional PID control algorithms have been compared. The course of
changes of power radiator \( (P_r) \) is presented in Fig. 2. The course of changes of power humidifier \( (P_{h2o}) \) is presented in Fig. 3. The course of changes of power humidifier \( (P_{h2o}) \) is presented in Fig. 4.

In all cases (in Fig. 2, 3 and 4) the dashed lines present the result of effect of classical algorithms – PID, the solid lines present the result of effect of predictive algorithms.

In the Fig. 5 there is shown the curve of PMV index changes during 24 hours for an laboratory room (continuous line – using the MPC algorithms; dashed line – using the classical control algorithms).

Figure 2: Changes of radiator power – PID algorithms (dashed line), MPC algorithms (solid line) – 15 September 2014

Figure 3: Changes of humidifier power – PID algorithms (dashed line), MPC algorithms (solid line) – 15 September 2014

Figure 4: Changes of cooler power – PID algorithms (dashed line), MPC algorithms (solid line) – 15 September 2014

Figure 5: Changes of PMV index – 15 September 2014

CONCLUSIONS

In the paper the behavior of the classical control systems and system using predictive control algorithms with model of room or HVAC in the room have been compared. PMV course of changes in both the control systems has been compared. Experiments were carried out in September 2014. The data of energy consumption of commercial HVAC systems for classic algorithms were from BMS. The data of energy consumption of HVAC systems for MPC algorithms were from simulation experiment using the room model. HVAC actuators for the control system of MPC algorithms use less energy than control system with PID algorithms. The course of the PMV indicator was closer to zero for MPC algorithms than the classical algorithms. For comparison algorithms we calculated the values: the total cost of the process (the sum of error control in the room, unit cost of poor air quality, airstream ventilation, enthalpy before and after the transformation of air, unit cost of air transformation, the cost of energy, pressure drop air in pipe, efficiency of actuators), cost of control expressed by the integral of the absolute control signal, control accuracy described by the integral of the absolute error signal and PMV indicator value for 24 hours expressed by the integral. The values: the total cost of the process (\( C_T \)), costs of control, values of the control accuracy for classical algorithms and prediction algorithm were compared (see Table 1).
Table 1: Comparison of algorithms - the costs and indicators

<table>
<thead>
<tr>
<th></th>
<th>PID</th>
<th>MPC</th>
</tr>
</thead>
<tbody>
<tr>
<td>C_T</td>
<td>129.0</td>
<td>112.7</td>
</tr>
<tr>
<td>Control cost</td>
<td>102.1</td>
<td>97.3</td>
</tr>
<tr>
<td>Control accuracy</td>
<td>85.5</td>
<td>82.1</td>
</tr>
<tr>
<td>PMV</td>
<td>0.21</td>
<td>0.08</td>
</tr>
</tbody>
</table>

REFERENCES


WEB REFERENCES

International Building Physics Toolbox in Simulink
(http://www.ibpt.org)
Accessed on 15/07/2015.

Accessed on 15/07/2015.

BIOGRAPHIES

Mariusz NOWAK was born in Poland and went to the Poznan University of Technology (control engineering). He obtained the PhD degree in 2007. From September 2007 he is an Assistant Professor at the Institute of Computing Science of the Poznan University of Technology. His research interest include: computer simulation, intelligent control system, computer control systems for environmental engineering, intelligent building systems, comfort climate control.

Andrzej URBANIAK was born in Poland and went to the Poznan University of Technology (control engineering) and Poznan University of A. Mickiewicz (mathematics). He obtained the PhD degree in 1979. From 1990 he is a professor of Institute of Computing Science. He is author or co-author of 5 books and over 200 papers concerning the computer control systems and application of computer science in environmental engineering.
ENGINEERING SIMULATION
CFD Analysis on Characteristics of CANDU6 Ventilation/Cooling Systems

S. Nicolici  
C.T. Mateescu  
Subsidiary of Technology and Engineering for Nuclear Projects (CITON-RATEN),  
Atomistilor Street, Magurele, Romania

R. Arsenescu  
S.C. Energonuclear S.A., 5-7 Vasile Lascăr,  
3rd floor, sector 2, Bucharest, Romania

I. Prisecanu  
D. Dupleac  
Power Engineering Faculty, Politehnica University of Bucharest  
313 Splaiul Independentei, Romania

KEYWORDS: Nuclear Engineering, CANDU reactor, Containment, CFD modeling, CFX.

Abstract: The aim of this paper is to determine the causes producing air overheating in the upper atmosphere of a generic CANDU containment by applying a numerical simulation approach in order to evaluate the natural / forced air convection. The phenomenon studied in this work is represented by single-phase fluid flow in a closed complex partitioned domain with conjugate heat transfer, determined by two driving forces: one produced by the cooling of the reactor building through local air coolers associated fans, and the other due to buoyancy effect created by the vertical temperature gradient. The results contain a series of sensitivity studies in relation to a series of input parameters: insulation properties, waste heat generation and cooling capacity.

Introduction

Using CFD (Computational Fluid Dynamics) codes for conducting detailed analyzes of specific nuclear reactor accident sequences recorded a significant progress in the past decade, mainly due to the extraordinary progress made in the areas of computer hardware and software (Smith, 2010). Although the efforts have been directed mainly in the direction of nuclear safety issues, demonstrated by many studies in this regard for validation of various physical models by experimental results (e.g., condensation, stratification, pressurized thermal shock, hydrogen management, etc.) (Andreani, et al., 2008), the use of these three-dimensional numerical tools for studying thermo-hydraulic phenomena related to the normal operation of power plant systems and equipment may prove as important in certain situations, such as thermally induced fatigue evaluation of piping systems, flow induced vibration, aging of concrete structures and equipment, and others.

For thermal-hydraulics that is reactor containment specific, the methodology for numerical analyses often involves the use of 0-D lumped parameters codes that provide volume averaged flow properties, for example hydrogen concentration or air temperature. But this approach represents an oversimplification because no spatial information is available and, where appropriate, phenomena that are inherently three-dimensional such as natural circulation or stratification in arbitrarily complex geometries is not well described. In such situations, as is the case presented in this work, the more accurate representation of the flow field by solving the governing equations in CAD created domains improve accuracy providing three-dimensional spatial distributions of the variables of interest (Houkema, Siccama, Lycklama a Nijeholt, & Komen, 2008).

Therefore, when a Nuclear Power Plant (NPP) from Romania equipped with CANDU reactor technology have been experiencing higher than normal temperatures in the upper part of the reactor containment during hot summer months, the approach to determine the causes and find solutions have relied almost exclusively on numerical simulations.

The aim of this study is to determine the causes producing air overheating in the upper atmosphere of a generic CANDU containment by using CFD simulations in order to evaluate the natural / forced air convection. The paper is structured as follows: Section 2 presents a short description of the CANDU containment, CAD development and investigated problem; the grid generation and testing, and modeling details are given in Section 3; further, in section “Results and Discussion” the results of the numerical simulations are given, including details of the flow (air temperature and streamlines) and parametric studies results; the last section summarizes the conclusions of the paper.

Problem specification

The reactor containment (or Reactor Building R/B) is provided by design with a ventilation/cooling system in order to remove waste heat from primary heat transport system components, moderator and piping systems. This cooling system is utilizing a series of air-water Local Air Coolers (LACs) and concrete fans (air–chilled water from a special system) to maintain the air temperature below an alarm limit of 49 °C (RAAN/CITON, 2005). However, due to (1) variation of the atmospheric conditions during the year, which in turn affects the thermal performances of the LACs, and (2) some design modification done without a detailed analysis of the systems involved, the temperatures measured by the reactor vault temperature loops have been reported to often exceed the limit even with all LACs in operation.

LACs are installed at various places within the R/B, providing cooling of the ambient air with forced air circulation (see Fig. 1). In most rooms of the R/B, one LAC is sufficient, but not in the fuelling-machine (F/M) and boilers rooms. These rooms, because of the presence of highly localized heat sources (e.g., end fittings, steam generators (SGs), primary pumps), require several LACs and associated distribution duct work (see Fig. 2-right). In the F/M rooms, the reactor vault concrete structure behind the feeder cabinets requires additional cooling during normal reactor operation. A concrete cooling fan in each F/M room draws air from the lower part of the F/M room through a heat exchanger cooled by chilled water, and discharges it through duct work, thereby cooling the reactor vault concrete surface behind the insulation cabinets (see Fig. 2-left).
The (R/B) CAD model was created starting from scratch in ANSYS Design-Modeler (SAS IP, 2013b), considering the real dimensions of the containment. Because of the great complexity of internal structures, platforms were developed one at a time starting from the base slab to the steam generator elevation. A series of CAD modeling simplifications were introduced in order to respond to the computing resources limits:

1) The R/B rooms included in the model are F/M, moderator and SGs rooms as seen in Fig. 1, and it is hypothesized that they do not change mass and energy with rest of the R/B rooms.

2) Among the existing equipment and components in the R/B volume were retained only those that could have a significant influence on air streamlines or are important heat sources. These are:
   - LACs and concrete coolers modeled as simplified rectangular domains having the same volume as the real setup;
   - The primary pumps representing an important source of heat;
   - The steam condenser-degasser (C-D), pressurizer (Pres.) and steam generators (SGs) which are the major heat sources;
   - Moderator and shutdown cooling heat exchangers introduced for hydraulic reasons.

3) The piping systems were not modeled on the assumption that they do not affect the airflow and are not major sources of heat (due to insulation).

4) The R/B dome was modeled explicitly (by introducing the concrete wall in the CAD model) aiming to evaluate the heat transferred to the upper water pool as accurately as possible. This water basin has a safety role, providing a cooling sink during accident scenarios.

5) The uncertainties related to the heat transfer at the outer surface of C-D, Pres. and SGs were minimized by introducing the insulation of these components in the CAD model. It has been assumed that the thermal resistances of the metal shells is zero, which is acceptable considering the much higher thermal conductivity of the steel compared to the one of insulation.

**CFD modeling**

The CFD model developed employing ANSYS CFX (SAS IP, 2013a) aimed at assessing the air temperature distribution in the entire R/B volume for different rates of heat losses. This variation in the input parameters (denoted as parametric studies) allowed us to understand what causes the mentioned problems: (1) insufficient shielding of the heat sources due to various components insulation degradation, or (2) insufficient capacity of the heat sinks caused by exterior atmospheric conditions (high hot summer temperature) and/or reduced thermal efficiency due to fouling on the cooling coil and flow reduction due to dirt accumulation in the LACs.

The research field of computational fluid dynamics is based on the basic conservation equations for mass, momentum and energy. By solving a set of nonlinear partial differential equations, more accurate results can be obtained, e.g. transient behavior, movement due to convection and transport due to laminar and turbulent diffusion (Anderson, 1995). ANSYS CFX uses a coupled solver, which solves the hydrodynamic equations (for \( u, v, w, p \)) as a single system. This solution approach uses a fully implicit discretization of the equations at any given time step. The spatial discretization algorithms and the transient formulation are presented in Table 1. For CFD simulations, RMS residual levels of 1E-4 could considered to be loosely converged, but in this paper other two criterions are used in order to achieve the right solution: domain imbalances and monitors. For turbulence modeling we have employed a two-equation model, \( k-e \), in which the turbulence kinetic energy scale is computed from the turbulent kinetic energy, which is provided from the solution of its transport equation. The turbulent length scale is estimated from two properties of the turbulence field, the turbulent kinetic energy and its dissipation rate. The dissipation rate of the turbulent kinetic energy is provided from the solution of its own transport equation.

<table>
<thead>
<tr>
<th>Table 1. Ansys CFD Solution methods</th>
</tr>
</thead>
<tbody>
<tr>
<td>Advection scheme</td>
</tr>
<tr>
<td>Turbulence numerics</td>
</tr>
<tr>
<td>Transient formulation</td>
</tr>
<tr>
<td>Solver precision</td>
</tr>
<tr>
<td>Timescale control</td>
</tr>
<tr>
<td>Converge criteria</td>
</tr>
</tbody>
</table>
CFD results can be strongly affected by the domain mesh size. Typically, sensitivity analyses of certain flow variables (mass flows, temperatures, pressures, etc.) in relation to a series of mesh grids with increasing number of elements are tested. These simulations are sometimes performed for a simplified flow domain and/or stationary regimes. The final discretization grid is chosen according to the required accuracy of the results versus available computational and time resources. Also, in certain circumstances one can compare the numerical results with experimental data or analytical solutions.

In order to obtain a grid invariant solution, a grid refinement analysis for three mesh grids (see Table 2) was performed: Mesh#1 (nodes 621k, elements 2,297k), Mesh#2 (nodes 998k, elements 3,615k) and Mesh#3 (nodes 1,748k, elements 5,741k). The obtained results (summarized in Fig. 2) show that the variation in terms of averaged heat fluxes and temperatures for various components is relatively small, the maximum difference between the results recorded for the mesh grids #1 and #2 is about 7.4% and between grids #2 and #3 the difference is less than 3.5%. From an engineering point of view, it was concluded that the solution is independent relative to grid # 3, which was considered acceptable for further analyses. Some views of the chosen grid are given in Fig. 4. It can be noticed that the insulations for each component have 4 elements across the thickness sufficient to capture the temperature gradient.

The average temperature is read at each LAC aspiration, being extracted from the output data at each iteration performed by the CFD solver;

- The maximum temperature difference inside each cooler is determined based on the water inlet temperature given as an input data depending on the water source (chilled water system or intermediate cooling water system);
- Having determined the (1) thermal efficiency rating of the LACs (evaluated considering the design values), (2) maximum temperature difference and (3) minimum thermal capacity, the thermal power transferred from the air to the cooling water is calculated at each iteration;
- Knowing the thermal power and the mass flow rates, the air outlet temperature is determined for each LAC, which is imposec as a boundary condition in the CFD model.

The efficiency of LAC can be calculated as the ratio of thermal power actually transferred and the maximum power that could be transferred if the surface of the countercurrent flow heat exchanger would be infinite:

$$\eta = \frac{P}{C_{\text{min}} \Delta T_{\text{max}}} = \frac{P}{\min(m \cdot c_p \cdot \Delta T) \cdot \frac{T_{\text{in}} - T_{\text{in, max}}}{T_{\text{in}} - T_{\text{in, min}}}}$$

where: $P$ = thermal power of the LAC at given conditions, $P_{\text{max}}$ = maximum power transferred at design conditions, $C_{\text{min}}$ = minimum thermal capacity, $\Delta T_{\text{max}}$ = maximum temperature difference, $m$ = mass flow rate, $c_p$ = specific heat at constant pressure, $T_{\text{in}}$ = inlet temperature.

The convergence of the numerical solutions was provided by achieving a low domain imbalance (less than 1%) for the conservation equations of mass and momentum and monitoring the temperatures in different areas of the R/B until a steady-state regime was obtained. We have noticed that a pseudo-transient simulation allowed a faster convergence than a stationary one since the time scale factors were user-controlled.

**Results and discussion**

The results of this paper were analyzed using the advanced post-processing capabilities of the ANSYS software. Temperature distributions, velocity vectors and streamlines on different R/B section planes are given in Figs. 5 and 6. The temperature on outer insulation surfaces for SGs, Pres. and C-D and on the upper concrete dome are represented in Figs. 7 and 8.
Air circulation in the reactor building has a complex behavior as revealed by the velocity vectors and streamlines. In the upper part of the R/B one can observe an ascending hot air current formed near the primary pumps, which in contact with the concrete dome produce a hot spot characterized by higher heat flux. Also, from the distributions of temperature and heat flux on the surface of the dome it is distinguished an asymmetry of the air flow produced most likely by the asymmetric positioning of some components (Pres. and C-D) and air distribution system. The temperature and heat flux maps on the components insulation show that distributions are highly uneven, with relatively large variations relative to the mean value. Existence of these hot areas is due to forced convection currents which change the axially symmetric distribution of a natural convection flow.
can have a significant (Smith, 2010) increase only in the following two cases:

- Variation of the primary pumps heat losses in the interval 0.6-1.5 x reference value,
- Reduction of the LACs thermal efficiency and air flow rate with 10 and 20% from the nominal values.

For comparison of the numerical results with NPP observations, the air temperature measurements during the summer period in 2011 and 2012, in the upper part of the containment are between 55 and 57 degrees C.

Conclusions

This paper presents a series of numerical simulations of air circulation in a CANDU600 reactor building, carried out in order to investigate the causes that lead to air overheating in the top region of the containment. The numerical procedure presented was carried on in accordance with paper's objective and comprised:

- Development of a simplified CAD model for the reactor building starting from the available documentation, aiming to include in the model only those components and structures important for the studied problem.
- Grid generation of the computing domain and mesh testing for numerical errors evaluation.
- Performing several parametric studies in order to determine the sensitivity of the results in relation to a series of input data.
- Post processing the simulation results and drawing the conclusions.

The analysis of the obtained results has led us to conclude that the R/B atmosphere overheating problem is most likely due to reduced performance of the local air coolers, which during the summer atmospheric condition cannot assure a proper heat sink for the waste energy generated inside the building.

References


MODELLING OF THE VERTICAL APPARENT MASS AT THE SEAT SURFACE AND BACKREST

Naser Nawaysheh
Mechanical Engineering Department,
College of Engineering, University of Sharjah,
P. O. Box 27272 Sharjah
United Arab Emirates
Email: nnawaysheh@yahoo.co.uk

KEYWORDS
Model, human body apparent mass, Backrest, vertical vibration

ABSTRACT

Many studies attempted to model the vertical apparent mass measured on the seat during vertical vibration. However, no work before has attempted to model the vertical apparent mass at the backrest during vertical vibration. The aim of this paper is to present a model that can predict the vertical apparent mass of the human body on the seat and at the backrest during whole-body vertical vibration. A two degree-of-freedom lumped parameter linear model was developed for that purpose. The model gave a close fit to both the median and individual responses of 12 male subjects. The model suggested that the resonance frequency in the response measured at the seat is produced by a translational vertical vibration mode which may indicate deformation of the tissue beneath the ischial tuberosities and/or the thighs. The model also suggested that the response measured at the backrest is produced not only by the direct interaction between the back and backrest but also by contribution from other body parts.

INTRODUCTION

Many studies have investigated the response of humans to whole-body vibration in the past 4 decades (e.g. Boileau and Rakheja 1998, Holmlund et al. 2000, Nawaysheh and Griffin 2005, Morioka and Griffin 2009). The nature of the investigations differed among the different studies depending on the goal of the study and the researchers’ interest. For example, some researchers were interested in investigating discomfort and interference with activities caused by vibration (e.g. Morioka and Griffin), some were interested in investigating vibration-health related issues (Bovenzi and Batta 1994), and others were interested in investigating the biodynamic responses to vibration (Nawaysheh and Griffin 2005). Understanding the biodynamic responses is essential for the interpretation of the results obtained from studies on discomfort, interference with activities and vibration-health related issues.

Biodynamic responses to vibration can be classified into two main frequency response functions. The first is called the mechanical impedance (or apparent mass) which describes the transmission of vibration to the human body or the gross response of the human body at the vibrating surface (Abdul Jalil and Griffin 2008). The second is called transmissibility and describes the transmission of vibration through the human body to different locations such as along the spine or the head (Paddan and Griffin 1998). Both response functions have been found experimentally to depend on the adopted posture and seating condition. For example, the apparent mass of a person sitting in a slouched body posture is different from that of the same person adopting an erect sitting posture (Fairley and Griffin 1989). The biodynamic responses to vibration have also been shown to be affected by using a backrest, a footrest and placing the hands on a steering wheel (Boileau and Rakheja 1998).

In real life situations of vibration exposure, the seated human body is more likely to be in contact with more than just the seat pan. The dependency of the biodynamic responses on the interactions between the body and supporting surfaces (e.g. backrest and footrest) implies the need to model these interactions when developing models representing the response of the human body to vibration.

The objective of this study is to present a mathematical quantitative model that can represent simultaneously the apparent mass of the human body measured on the seat surface and at the backrest during vertical vibration. The model will be used to represent the median and the individual data of 12 subjects. Sensitivity test will be conducted to identify the parameters of the model contributing to the resonances appearing in the experimentally measured apparent mass. It was hypothesized that a two degree-of-freedom model will be sufficient to represent the moduli of the measured apparent masses.

THE EXPERIMENTAL DATA

The proposed model will be used to represent the vertical apparent mass on the seat surface and at the backrest measured simultaneously during vertical vibration. The vertical apparent masses of 12 subjects were measured with the subjects sitting on a flat rigid seat with a vertical backrest that moved in phase with the seat. The subjects rested their feet on a footrest that also moved in phase with the seat such that their upper legs were horizontal and their
lower legs were vertical. The vibration was random with a magnitude of 1.25 ms$^{-2}$ r.m.s in the frequency range 0 to 20 Hz. The magnitude of the vertical apparent mass measured at the backrest was much less than the magnitude of the apparent mass measured on the seat. However, including the response at the backrest when modelling the response of the human body to vibration could advance our understanding of how the human body moves under vibration. More details of the experimental setup can be found in (Nawayseh and Griffin 2004).

THE PROPOSED MODEL.

The model used in this study is a quantitative two-degree-of-freedom lumped parameter linear model (Figure 1). The model has two masses (mass 1 and mass 2) and has only translational capability. The interaction between the body and the seat pan was modelled by a linear vertical spring and linear vertical damper. Similarly, the interaction between the back and the backrest was modelled by a linear vertical spring and linear vertical damper. Since the proposed model is not a mechanistic model, none of the parameters of the model is intended to represent any specific part of the human body.

The mathematical equations of the model

The following equations of motion of the model were derived using Newton’s second law, $\sum F=ma$, where $F$ is the force(s) acting on the object, $m$ is the mass of the object and $a$ is the acceleration of the object:

$$m_1 \frac{d^2z_1}{dt^2} + c_1 \left( \frac{dz_1}{dt} + \frac{dz_2}{dt} \right) + k_1 (z_1 - z_b) + k_2 (z_1 - z_2) = 0$$

$$m_2 \frac{d^2z_2}{dt^2} + c_2 \left( \frac{dz_1}{dt} + \frac{dz_2}{dt} \right) + c_1 \left( \frac{dz_1}{dt} + \frac{dz_2}{dt} \right) + k_1 (z_1 - z_b) + k_2 (z_2 - z_b) = 0$$

The motions of both mass 1 and mass 2 contribute to the vertical force at the seat while only mass 2 contributes to the vertical force at the backrest. The equations for the vertical force at the seat ($f_s(t)$) and the vertical force at the backrest ($f_b(t)$) can be written as:

$$f_s(t) = m_2 \frac{d^2z_2}{dt^2}$$

$$f_b(t) = m_2 \frac{d^2z_2}{dt^2}$$

Where, $m_1$ and $m_2$ are the masses of mass 1 and mass 2, respectively.

$k_1$ and $c_1$ are the vertical stiffness and damping coefficient beneath mass 1 which represent the interaction between the body and the seat surface.

$k_2$ and $c_2$ are the vertical stiffness and damping coefficient between mass 1 and mass 2.

$k_1$ and $c_1$ are the vertical stiffness and damping coefficient that represent the interaction between the back of the subject and the backrest.

$z_1$ represents the motion of mass 1

$z_2$ represents the motion of mass 2

$z_b$ represents the vertical motion of the seat base and backrest where the force and acceleration used to calculate the apparent mass were measured.

The apparent mass can be calculated by finding the Laplace transform of the above equations assuming zero initial conditions and solve for the complex ratio between $F_s(s)$ or $F_b(s)$ and $s^2Z_d(s)$ where $F_s(s)$ is the Laplace transform of the force on the seat, $f_s(t)$, $F_b(s)$ is the Laplace transform of the force at the backrest, $f_b(t)$, $s^2Z_d(s)$ is the Laplace transform of the acceleration of the seat surface and backrest, $\ddot{z}_d(t)$, and $s$ is $j\omega$ where $j$ is $\sqrt{-1}$ and $\omega$ is the angular frequency in rad/s. The apparent mass on the seat (AMS) and the apparent mass at the backrest (AMB) can be found from the following equations:

$$AMS = m_1 \left[ \frac{C + B}{A + AD - B^2} \right] + m_2 \left[ \frac{A + C B}{A + AD - B^2} \right]$$

$$AMB = m_2 \left[ \frac{A + C B}{A + AD - B^2} \right]$$

Where,

$A = m_1 s^2 + c_1 s + c_2 s + k_1 + k_2$

$B = c_2 s + k_2$

$C = c_1 s + k_1$

$D = m_2 s^2 + c_2 s + c_1 s + k_1 + k_2$

$E = c_1 s + k_1$

The moduli of the apparent mass on the seat and the apparent mass at the backrest can be obtained from the real and imaginary parts of the complex apparent masses, AMS and AMB:

$$\text{AMS}_{\text{mod}}(\omega) = \sqrt{(\text{Re} \text{AMS}(\omega))^2 + (\text{Im} \text{AMS}(\omega))^2}$$

$$\text{AMB}_{\text{mod}}(\omega) = \sqrt{(\text{Re} \text{AMB}(\omega))^2 + (\text{Im} \text{AMB}(\omega))^2}$$
Optimisation of the parameters of the model

A total of eight model parameters were optimized \((m_1, m_2, c_1, c_2, c_3, k_1, k_2, \text{ and } k_3)\) using the interior-point algorithm within MATLAB. The function \textit{fmincon} was used for this purpose as it allows for constraining the parameters to have values within a certain range. This is useful to avoid having a negative value for the optimized parameters. The optimised parameters were obtained by reducing the error between the moduli of the vertical apparent masses on the seat and at the backrest measured by Nawayseh and Griffin 2004 and the vertical apparent masses on the seat and at the backrest calculated from the model. So, the following error function was used for that purpose:

\[

er = \sum_{\text{s}} \left( \text{AM}S^\text{mod, s} - \text{AM}S^\text{mod, m} \right)^2 + \sum_{\text{b}} \left( \text{AM}B^\text{mod, s} - \text{AM}B^\text{mod, m} \right)^2
\]

where,

\(\text{AM}S^\text{mod, s}\) and \(\text{AM}S^\text{mod, m}\) are the moduli of the calculated and measured vertical apparent masses on the seat, respectively, \(\text{AM}B^\text{mod, s}\) and \(\text{AM}B^\text{mod, m}\) are the moduli of the calculated and measured vertical apparent masses at the backrest, respectively, and \(w\) is an arbitrary weighting factor employed to improve the prediction. This factor was needed as the order of magnitude of the apparent mass on the seat is greater than that at the backrest.

RESULTS AND DISCUSSION

Prediction of Median data

The model was used to represent the modulus of the median vertical apparent mass of 12 subjects measured on the seat surface and at the backrest (Figure 2). The figure shows that the model is able to provide a close fit to the moduli of the median vertical apparent mass both on the seat and at the backrest. The parameters of the model obtained by fitting the median data are given in the last row of Table 1. The seat-person system is a coupled system in which the dynamic response of the seat affects the biodynamic response of the seat occupant and the biodynamic response of the occupant of the seat affects the response of the seat including its vibration transmissibility. This is true for both the seat surface and the backrest. The optimised parameters that were obtained from fitting the median data can be very useful in evaluating the vertical transmissibility of vehicle seats at both the seat surface and backrest levels during vertical vibration. This can be achieved by combining the proposed model with a mathematical model of the seat itself to produce one coupled system used to evaluate the transmissibility of the seat pan and seat backrest (Wei 2000; Qiu and Griffin 2011). Alternatively, the parameters of the model can be used for building anthropodynamic dummy that can be supported on the seat surface and backrest to evaluate the performance of seats exposed to vibration instead of using human subjects. This will have the advantage of obtaining repeatable measurements as well as the possibility of testing the performance of the vehicle seat under extreme vibration conditions.

Some researchers used a single degree of freedom mass-spring-damper model to represent the vertical apparent mass measured at the seat surface (Toward and Griffin 2010). It has also been shown that using a two-degree-of-freedom model improves the model representation of the vertical apparent mass measured on the seat surface. In the current work, a single-degree-of-freedom model cannot be used to represent simultaneously the vertical apparent mass on the seat surface and the vertical apparent mass at the backrest. This is due to the structure of the single-degree-of-freedom system which has only one mass: with one mass and with the backrest moving with the same motion as that of the seat, it is not possible to have a coupling between the mass and the seat different from that between the mass and the backrest.

The two-degree-of-freedom models reported previously did not account for the interaction with the backrest (Wei 2000). In fact, they were developed to represent the vertical apparent mass measured on a seat without a backrest. Since models that represent the vertical apparent mass on a seat with a backrest are not available in the literature (even without modelling the interaction between the body and backrest), it was not possible in this current work to identify the effect of modelling the interaction with the backrest on the parameters of the model \((m_1, m_2, k_1, c_1, k_2, c_2)\). This could be performed in future work by repeating the current study but with ignoring the interaction with the backrest and investigating the changes to the parameters of the model.

Prediction of individual data

The model was employed to represent the moduli of the vertical apparent masses of 12 subjects measured on the seat and at the backrest. The calculated (i.e. from the model) and measured (i.e. from the experiment) apparent masses were in good agreement for both the moduli of the vertical apparent mass on the seat (Figure 3) and the moduli of the vertical apparent mass measured at the backrest (Figure 4). The parameters of the model obtained for each subject are shown in Table 1. The table shows high variability in the parameters among subjects. This is not surprising given the high inter-subject variability shown in the experimental data (Figure 5) which was produced by using subjects with different mass, stature, body mass index (BMI) and age.
The average coefficient of variation (CV) indicated greater inter-subject variability in the vertical apparent mass measured at the backrest (CV=0.393) than in the vertical apparent mass measured at the seat surface (CV=0.197). One more factor that may have added to the high inter-subject variability in the vertical apparent mass at the back is the different contact location between the back and backrest among the subjects: relative to their backs, tall subjects had low contact point between their backs and the backrest while short subjects had high contact point between their backs and the backrest. A previous study had shown that the forces at the back were highly dependent on the location of the measurement (Abdul Jalil and Griffin 2008). Although the study by Abdul Jalil and Griffin was under fore-and-aft vibration, one could hypothesize that similar results will be observed at the back during vertical vibration.

<table>
<thead>
<tr>
<th>Sub. No.</th>
<th>$m_1$ (kg)</th>
<th>$m_2$ (kg)</th>
<th>$k_1$ (N/m)</th>
<th>$k_2$ (N/m)</th>
<th>$k_3$ (N/m)</th>
<th>$c_1$ (N.s/m)</th>
<th>$c_2$ (N.s/m)</th>
<th>$c_3$ (N.s/m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>51</td>
<td>1.0</td>
<td>70461</td>
<td>1313</td>
<td>697</td>
<td>2138</td>
<td>6.4</td>
<td>14.5</td>
</tr>
<tr>
<td>2</td>
<td>63</td>
<td>2.6</td>
<td>65689</td>
<td>60590</td>
<td>0.0006</td>
<td>1708</td>
<td>0.0002</td>
<td>93.9</td>
</tr>
<tr>
<td>3</td>
<td>59</td>
<td>2.0</td>
<td>87119</td>
<td>6023</td>
<td>1337</td>
<td>1823</td>
<td>2.2E-6</td>
<td>50.4</td>
</tr>
<tr>
<td>4</td>
<td>50</td>
<td>1.5</td>
<td>67859</td>
<td>1802</td>
<td>1781</td>
<td>63.1</td>
<td>1E-6</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>65</td>
<td>2.6</td>
<td>80828</td>
<td>49789</td>
<td>0.0009</td>
<td>1632</td>
<td>3.3E-5</td>
<td>143.1</td>
</tr>
<tr>
<td>6</td>
<td>53</td>
<td>1.6</td>
<td>56883</td>
<td>7148</td>
<td>0.0001</td>
<td>1575</td>
<td>8.0</td>
<td>38.4</td>
</tr>
<tr>
<td>7</td>
<td>49</td>
<td>1.5</td>
<td>76482</td>
<td>3.1E5</td>
<td>0.0042</td>
<td>0.0020</td>
<td>380.4</td>
<td>1625.1</td>
</tr>
<tr>
<td>8</td>
<td>45</td>
<td>1.4</td>
<td>54071</td>
<td>0.0063</td>
<td>1600</td>
<td>1176</td>
<td>101</td>
<td>2.3E-5</td>
</tr>
<tr>
<td>9</td>
<td>49</td>
<td>2.1</td>
<td>38031</td>
<td>4.3E5</td>
<td>0.0171</td>
<td>0.0009</td>
<td>0.0005</td>
<td>1271</td>
</tr>
<tr>
<td>10</td>
<td>51</td>
<td>4.1</td>
<td>55563</td>
<td>1.9E5</td>
<td>0.0030</td>
<td>893</td>
<td>816.0</td>
<td>781.1</td>
</tr>
<tr>
<td>11</td>
<td>66</td>
<td>1.5</td>
<td>78314</td>
<td>4.6E-6</td>
<td>2173</td>
<td>2416</td>
<td>24.1</td>
<td>3.4E-8</td>
</tr>
<tr>
<td>12</td>
<td>84</td>
<td>2.0</td>
<td>61427</td>
<td>32602</td>
<td>0.0007</td>
<td>1823</td>
<td>320.2</td>
<td>142.7</td>
</tr>
<tr>
<td>Med</td>
<td>55.2</td>
<td>1.6</td>
<td>67046</td>
<td>0.0066</td>
<td>1563</td>
<td>1946</td>
<td>61</td>
<td>4.0E-5</td>
</tr>
</tbody>
</table>

The coefficient of variation was found greater for $m_2$ than for $m_1$. The coefficient of variation was also found greater for the stiffness and damping coefficients of the springs and dampers connected to mass 2 (i.e. $k_2$, $k_3$, $c_2$ and $c_3$) than the stiffness and damping coefficient of the spring and damper connected to mass 1 only (i.e. $k_1$ and $c_1$). This is consistent with higher inter-subject variability in the vertical apparent mass at the backrest than in the vertical apparent mass at the seat found experimentally and mentioned above. To explore this further, sensitivity analysis was performed to see the effect of changing the model parameters on the responses.

**Sensitivity analysis**

The optimised parameters were used to identify the contribution of each parameter to the two responses. This was performed by allowing the parameters to vary by ±40% of their optimised values and observe any changes in the responses. The studied parameters were $k_1$, $k_2$, $k_3$, $c_2$ and $c_3$. The sensitivity tests performed on the parameters of the median responses are shown in Figure 6 for the vertical apparent mass at the seat surface and Figure 7 for the vertical apparent mass at the backrest.

Figure 6 shows the changes in $k_1$ and $c_1$ only affected the median vertical apparent mass at the seat while changes in the other parameters (i.e. $k_2$, $c_2$, $k_3$ and $c_3$) had no influence on the median vertical apparent mass on the seat. Changes in $k_1$ and $c_1$ resulted in changes in the median vertical apparent mass on the seat over almost the whole frequency range but greatly around the resonance region. So, although the proposed model is a very simple non-mechanistic model, this result may indicate that the peak of the vertical apparent mass on the seat is produced by deformation of the tissue beneath the ischial tuberosities and/or beneath the thighs consistent with previous suggestions (Sandover 1978, Kitazaki 1994, Nawayseh and Griffin 2003).
CONCLUSIONS

A two degree-of-freedom lumped parameter linear model was able represent the moduli of the vertical apparent mass measurec on the seat surface and the vertical apparent mass measurec at the backrest during vertical vibration. The model gave close fit to both the median and individual responses of 12 male subjects. The model suggested that the resonance frequency in the response measured at the seat is produced by a translational vertical vibration mode which may indicate deformation of the tissue beneath the ischial tuberosities and/or the thighs. The model also suggested that the response measured at the backrest is produced not only by the direct interaction between the back and backrest but also by contribution from other body parts.

REFERENCES


Figure 6 Changes in the Apparent Mass on the Seat Resulting from Changes in the Optimised Model Parameters.

Figure 7 Changes in the Apparent Mass at the Backrest Resulting from Changes in the Optimised Model Parameters.

Figure 7 indicates that the vertical apparent mass at the backrest is affected by changes in the stiffness and damping coefficient beneath mass 1 (i.e. $k_1$ and $c_1$), the damping coefficient between mass 1 and mass 2 (i.e. $c_2$) and the stiffness at the interaction between mass 2 and the backrest (i.e. $k_2$). This implies that the stiffness and damping coefficient beneath mass 1 contribute to the resonance frequency of the vertical apparent mass at the backrest.

As mentioned above, the experimental results showed less variability in the apparent mass on the seat surface than in the apparent mass at the backrest. This is reflected in the optimised parameters: the variability in the optimised values of $k_1$ and $c_1$, which were the only contributors to the apparent mass on the seat surface, is much less than the variability in the optimised values of the other parameters which contributed to the apparent mass at the backrest.

The parameters of the model presented in this paper are applicable to the sitting posture and the vibration magnitude used in this study. More work should be carried out to identify the parameters of the model in other postures such as that adopted by car drivers. The mathematical models of the apparent mass of the body can be used to predict seat transmissibilities or design seats that attenuate or prevent the vibration from reaching the human body through the seat surface and/or backrest.
AUTHOR BIOGRAPHY

NASER NAWAYSEH was born in Jordan and studied Mechanical Engineering at the University of Jordan and graduated in 1993. He obtained his MSc by research in Solar Water Desalination in 1996 and PhD in biodynamic responses to vibration from the University of Southampton in The United kingdom in 2004. He worked as a Research Fellow at the Institute of Sound and Vibration Research for 3 years. He then joined Dhofar University in Oman and worked as an Assistant Professor and Chairperson of the Mechanical and Mechatronics Engineering Department between 2007 and 2013. He joined the University of Sharjah in the United Arab Emirates as an Associate Professor in September 2013.
Mathematical Analysis on Pitch and Roll Variations Based Hit Probability Prediction for a Combat Vehicle Design

Hwan Il Kang∗
Hyun Soo Kim∗

Department of {Information & Communications Eng. ∗∗}
Myongji University
Yongin, South Korea
E-mail: hwanilkang@naver.com∗
saints7777@naver.com∗

KEYWORDS
Hit probability, Vertical and horizontal variations, Pitch variation, Roll variation, Combat vehicle, Error estimates

ABSTRACT

When a combat vehicle runs on an uneven road, the combat vehicle has some error budgets such as vertical displacement, pitch variation and roll variation. These error budgets result in the dispersion of bullets on the target. The pitch and roll variation results in the variations of the elevation and azimuth angles. In this paper, the relationships between the pitch and roll variations and the hit probability are obtained under the assumption that the fixed biased errors can be fixed by the inertial navigation system. From the trajectory differential equations, we obtain the vertical and horizontal variations for the bullet at some known distance. From the Gaussian distribution of the dispersion of bullet at the target, the hit probability can be obtained. In addition, the vertical and horizontal distance variations at the target may be obtained by the given pitch and roll variations. We show the usefulness of our analysis by two simulation experiments. The hit probabilities based on mathematical analysis are nearly equal to those which come from the simulation results.

INTRODUCTION

When we design a four-wheeled or six-wheeled combat vehicle, four important aspects are mobility, fire power, vulnerability and operability of the combat vehicle. Mobility is concerned with how fast it drives on the wild road; Fire power is related to fire performances of arms equipped with combat vehicle; Vulnerability is concerned with robustness against enemy fire powers; Operability refers to human factors of combat vehicle. Among four performances of combat vehicle, the fire power performance plays the most important role in the combat environment. The important index of fire power performance is a hit probability. The fire power performance for combat vehicle with suspension system has been studied in the paper (Kang et al. 2015c). Hit probability is a function of error budgets (Weaver 1990; Groves 1963). Error budgets come from errors of subsystems. Errors of subsystem are divided into five categories: errors from internal ballistics, mechanical errors from gun and turret, trajectory errors from external ballistics, meteorological errors of sensors, errors of combat vehicle with road conditions such as gun pitch angle error, gun roll angle error, gun vertical displacement error. Table 1 tabulates subsystems and related errors of combat vehicles (Kang et al. 2015b).

<table>
<thead>
<tr>
<th>Subsystems</th>
<th>Kinds of errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal ballistics</td>
<td>error of mass of propellant, muzzle velocity error</td>
</tr>
<tr>
<td>gun and turret</td>
<td>aiming error (pitch angle error) mechanical error</td>
</tr>
<tr>
<td>external ballistics</td>
<td>trajectory error</td>
</tr>
<tr>
<td>meteorological sensors</td>
<td>temperature error, density error</td>
</tr>
<tr>
<td>Combat vehicles with road conditions</td>
<td>gun pitch angle error, gun roll error, gun vertical displacement error</td>
</tr>
</tbody>
</table>

Hit probability depends on the errors from subsystems of combat vehicles. The target is located in some distance from the position of combat vehicle. On the target, the dispersion of the bullets exists due to the errors from subsystems of combat vehicle. In this paper, we focus on the pitch error and roll error for the combat vehicle and the pitch error and roll error result in the dispersion on the target. This dispersion may be obtained from the trajectory differential equations. From Gaussian distribution of the dispersion on the target, we obtain the hit probability associated with the pitch error and roll error. Finally, we show the usefulness of our analysis by two simulation experiments. The hit probabilities based on mathematical analysis are nearly equal to those which come from the simulation results.
ERROR ESTIMATES AND HIT PROBABILITY PREDICTION

There are two error estimates methods: one is the engineering based method and the other is the experimental based method. These have been described in paper (Kang et al. 2015b). We use the experimental approach as the error estimates. The experimental approach uses the unit partial and estimation of total errors. The total errors of gun pitch angle and gun roll angle are transferred to the total errors at the target through the trajectory equation. In addition, there are two methods of hit probability prediction: one is an analytic method and the other is a Monte Carlo method. These have been described in paper (Kang et al. 2015b). In this paper, hit probability prediction may be obtained using both the analytic method and Monte Carlo method. The analytic method assumes that hit probability is expressible as the integral of Gaussian distribution of errors at the target.

TRAJECTORY DIFFERENTIAL EQUATIONS

The trajectory equations (McCoy, 1999) are expressed as follows:

\[
\begin{align*}
\dot{v}_x &= -C_{d,i} \bar{V}(v_x - w_x) \\
\dot{v}_y &= -C_{d,i} \bar{V}(v_y - w_y) - g \\
\dot{v}_z &= -C_{d,i} \bar{V}(v_z - w_z)
\end{align*}
\]  

(1)

where \( \bar{V} = \sqrt{\sum(v_i - w_i)^2} \). The x-axis is a range direction and the y-axis is a upward direction against the gravitational force and z-axis is a deflection direction. The trajectory of a bullet is shown in Figure 1. The approximate trajectory of bullet was studied in the paper (Kang et al. 2015a).

![Bullet Trajectory](image)

Figure 1: Bullet Trajectory

HIT PROBABILITY

When a bullet hits a target, the dispersion of the bullet forms a Gaussian distribution whose center is a center of the target. The height and the width of the target are assumed to be \( 2a \), \( 2h \), respectively. Hit probability (MacFadzean 1992) is defined as

\[
p_{sh} = \frac{1}{2\pi\sigma_x\sigma_y} \int_{(x,y) \in \text{Target}} \exp \left\{ -\frac{(x-h)^2}{2\sigma_x^2} \right\} \exp \left\{ -\frac{(y-k)^2}{2\sigma_y^2} \right\} dx dy
\]

(2)

where \(-a \leq x \leq a, -b \leq y \leq b\). Hit probability is expressible using the error function as

\[
p_{sh} = \frac{a}{\sqrt{2\pi}} \int_0^a e^{-t^2} dt + \frac{b}{\sqrt{2\pi}} \int_0^b e^{-t^2} dt
\]

(3)

Using the error function

\[
\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt
\]

(4)

Equation (3) is rewritten as

\[
p_{sh} = \sigma_f \left[ \frac{a}{\sqrt{2\sigma_h}} \right] \sigma_f \left[ \frac{b}{\sqrt{2\sigma_y}} \right]
\]

(5)

ROLL VARIATION

When the vehicle stops, the direction of gun is assigned to the range direction (the positive X axis). When a combat vehicle runs on an uneven road, the vehicle has a roll variation. The geometrical diagram is depicted in Figure 2. The line \( \overline{AO} \) is a tilted gun resulting from the roll angle denoted by \( \theta_r \), depicted in Figure 2. Since \( \overline{AB} = \overline{BO}\tan(\varphi) \) and \( \overline{BB} = \overline{AB}\sin(\theta_r) \), we obtain

\[
\tan(\varphi) = \frac{\overline{BB}}{\overline{BO}} = \sin(\theta_r)\tan(\varphi)
\]
and taking the inverse of the tangent, the azimuth angle ($\phi$) is expressible as

$$\phi = \tan^{-1}(\sin(\theta_r)\cos(\beta))$$  \hspace{1cm} (6)

where the angle ($\beta$) is a gun direction from the horizontal plane (Groves 1963). Taking its differentiation with respect to $\phi$, the azimuth variation with respect to the change of roll angle is obtained by

$$\frac{\partial \phi}{\partial \theta_r} = \frac{\cos(\theta_r)\tan(\beta)}{1 + \tan^2(\beta)\sin^2(\theta_r)}$$  \hspace{1cm} (7)

Since $\overline{AO} = \overline{BO}/\cos(\beta)$ and $\overline{DO} = \overline{DO}/\cos(\phi)$, we obtain

$$\cos(\beta) = \frac{\overline{DO}}{\overline{AO}} = \frac{\alpha s}{\beta s} (\beta)$$

and taking the inverse of the cosine

$$\theta_e = \alpha s^{-1}(\beta s)/(\beta s)$$  \hspace{1cm} (8)

Taking its differentiation with respect to $\phi$, we obtain

$$\frac{\partial \theta_e}{\partial \phi} = \frac{-\alpha s (\beta) \sin(\phi)}{\alpha s^2(\phi) \sqrt{1 - \alpha s^2(\beta)/\alpha s^2(\phi)}}$$  \hspace{1cm} (9)

By the chain rule, the elevation variation with respect to the change of roll angle is expressible as

$$\frac{\partial \theta_e}{\partial \phi} \frac{\partial \phi}{\partial \theta_r} = \frac{-\alpha s (\beta) \sin(\phi) \cos(\theta_r)}{\alpha s^2(\phi)(1 + \alpha s^2(\beta)/\alpha s^2(\phi)) \sqrt{1 - \alpha s^2(\beta)/\alpha s^2(\phi)}}$$  \hspace{1cm} (10)

ELEVATION AND AZIMUTH VARIATIONS DUE TO PITCH AND ROLL ANGLES

In the previous section, we assume that the gun tilted angle $\beta$ is fixed. But, in real situations, the gun tilted angle $\beta$ is not fixed. The gun tilted angle $\beta$ is changed to the time varying angle $\beta + \theta_p$ where the mean value of $\theta_p$ is equal to zero and its standard deviation is $\sigma_{\theta_p}$. We are in a position to present an elevation and azimuth variations if the gun tilted angle $\beta + \theta_p$ and the roll angle $\theta_r$ is given. In this case, we obtain

$$\phi = \tan^{-1}(\sin(\theta_r)\tan(\beta + \theta_p))$$  \hspace{1cm} (11)

Taking its differentiation with respect to $\phi$, the azimuth variation with respect to the change of roll angle is obtained by

$$\frac{\partial \phi}{\partial \theta_r} = \frac{\cos(\theta_r)\tan(\beta + \theta_p)}{1 + \tan^2(\beta)\sin^2(\theta_r)}$$  \hspace{1cm} (12)

Taking its differentiation with respect to $\phi$, the azimuth variation with respect to the change of tilted angle is obtained by

$$\frac{\partial \phi}{\partial \theta_p} = \frac{\sin(\theta_r)(1 + \tan^2(\beta + \theta_p))}{1 + \tan^2(\beta + \theta_p)\sin^2(\theta_r)}$$  \hspace{1cm} (13)

Similarly, the elevation angle is given by

$$\theta_e = \alpha s^{-1}(\beta s)/(\beta s)$$  \hspace{1cm} (14)

Taking its differentiation with respect to $\phi$, we obtain

$$\frac{\partial \theta_e}{\partial \phi} = \frac{-\alpha s (\beta + \theta_p) \sin(\phi)}{\alpha s^2(\phi) \sqrt{1 - \alpha s^2(\beta + \theta_p)/\alpha s^2(\phi)}}$$  \hspace{1cm} (15)

Taking its differentiation with respect to $\beta$, we obtain

$$\frac{\partial \theta_e}{\partial \beta_p} = \frac{\sin(\beta + \theta_p)}{\alpha s^2(\phi) \sqrt{1 - \alpha s^2(\beta + \theta_p)/\alpha s^2(\phi)}}$$  \hspace{1cm} (16)

Taking the differentiation of $y$ with respect to $\theta_p$

$$\frac{\partial y}{\partial \theta_p} = \frac{\partial y}{\partial \theta_e} \frac{\partial \theta_e}{\partial \theta_p}$$  \hspace{1cm} (17)

where $\frac{\partial y}{\partial \theta_e}$ can be calculated using the trajectory equations.

Taking the differentiation of $y$ with respect to $\theta_r$

$$\frac{\partial y}{\partial \theta_r} = \frac{\partial y}{\partial \theta_e} \frac{\partial \theta_e}{\partial \theta_r}$$  \hspace{1cm} (18)

where $\frac{\partial y}{\partial \theta_e}$ can be calculated using the trajectory equations.

Similarly, we obtain

$$\frac{\partial x}{\partial \theta_p} = \frac{\partial x}{\partial \theta_e} \frac{\partial \theta_e}{\partial \theta_p}$$  \hspace{1cm} (19)

and

$$\frac{\partial x}{\partial \theta_r} = \frac{\partial x}{\partial \theta_e} \frac{\partial \theta_e}{\partial \theta_r}$$  \hspace{1cm} (20)

The hit probability due to pitch and roll variations is given by
\[ p_{\text{hit}} = \sigma_f \left( \frac{a}{\sqrt{2} \sigma_x} \right) \sigma_f \left( \frac{b}{\sqrt{2} \sigma_y} \right) \]  

(21)

where \(2a\) and \(2b\) are width and height of the target, respectively. In addition, the variance of the vertical displacement is given by

\[ \sigma_y^2 = \left( \frac{\partial y}{\partial \theta_p} \right)^2 \sigma_{\theta_p}^2 + \left( \frac{\partial y}{\partial \theta_r} \right)^2 \sigma_{\theta_r}^2, \]  

(22)

and the variance of the horizontal displacement is given by

\[ \sigma_x^2 = \left( \frac{\partial x}{\partial \theta_p} \right)^2 \sigma_{\theta_p}^2 + \left( \frac{\partial x}{\partial \theta_r} \right)^2 \sigma_{\theta_r}^2, \]  

(23)

**MATHEMATICAL RESULTS AND MONTE CARLO SIMULATIONS**

Throughout this paper, we assume that the muzzle velocity is \(v_m = 241\) [m/sec], the range is 700[m] the modified drag coefficient is \(C_D = 5.1313e-4\) [m^{-1}]. We assume that the fixed biased errors can be fixed by the inertial navigation system, too. Suppose that \(\phi = 5^o, \theta_p = 0.02\) [radian] and \(\theta_r = 0.0\) [radian]. In addition, \(\sigma_{\theta_p} = 0.5\) [mradian] and \(\sigma_{\theta_r} = 1.7\) [mradian] are given. By (15) and (16), we obtain

\[ \frac{\partial y}{\partial \theta_p} = (0.6108)(1) = 0.6108 \left[ \frac{\text{m}}{\text{mrad}} \right] \]

and

\[ \frac{\partial y}{\partial \theta_r} = (0.6108)(0.0017) = 0.0010 \left[ \frac{\text{m}}{\text{mrad}} \right] \]

Similarly, equations (17) and (18) give us

\[ \frac{\partial x}{\partial \theta_p} = (0.6997)(0.0202) = 0.0141 \left[ \frac{\text{m}}{\text{m rad}} \right] \]

and

\[ \frac{\partial x}{\partial \theta_r} = (0.6997)(0.0875) = 0.0612 \left[ \frac{\text{m}}{\text{m rad}} \right] \]

Finally, supposing that the height and width of the target are 4m and 4m, respectively, then the hit probability is

\[ p_{\text{hit}} = \sigma_f \left( \frac{2}{\sigma_y \sqrt{2}} \right) \sigma_f \left( \frac{2}{\sigma_x \sqrt{2}} \right) = 94.59\% \]  

(24)

where \(\sigma_x = 0.0389\) [m] and \(\sigma_y = 1.384\) [m] are given by equations (22) and (23). By Monte Carlo method, the dispersion pattern at the target is depicted in Figure 4. The calculated hit probability is 93.01\%. This hit probability is almost the same as the hit probability obtained analytically in (24) which is 94.59\%. The discrepancy is due to the interaction between the pitch and roll angles. The interaction between them cannot be measured in the analytic way.

**Figure 4:** Bullet Dispersion Pattern When Standard Deviations of Pitch and Roll Angles are 1.7 and 0.5 [mrad], respectively. Hit Probability is 93.01\%

For the second example, assume that \(\sigma_{\theta_p} = 3\) [mrad] and \(\sigma_{\theta_r} = 2\) [mrad]. In this case, hit probability is given by

\[ p_{\text{hit}} = \sigma_f \left( \frac{2}{\sigma_y \sqrt{2}} \right) \sigma_f \left( \frac{2}{\sigma_x \sqrt{2}} \right) = 89.84\% \]  

(25)

where \(\sigma_x = 0.0389\) [m] and \(\sigma_y = 1.2216\) [m] are given by equations (22) and (23).

**Figure 5:** Bullet Dispersion Pattern When Standard Deviations of Pitch and Roll Angles are 2.0 and 3.0 [mrad], respectively. Hit Probability is 87.11\%

By Monte Carlo method, the dispersion pattern at the target is depicted in Figure 5. The calculated hit probability is 87.11\%. This hit probability is almost the same as the hit probability...
obtained analytically in (25), which is 89.84%.

CONCLUSIONS

In this paper, when a combat vehicle runs on an uneven road, pitch and roll variations of the vehicle occur. These pitch and roll variations result in variations of the elevation and azimuth angles. Using the bullet trajectory, we obtained the vertical distance variation and azimuth distance variations with respect to the elevation angle variation and azimuth angle variation. At the target, for error estimates, we use an experimental approach. Finally, at the target, hit probability with respect to the variations of pitch and roll angle can be predicted, by using both an analytic method and Monte Carlo method. We have shown the usefulness of our analysis by two simulation experiments. The hit probabilities based on mathematical analysis are nearly equal to those which come from the simulation results.

ACKNOWLEDGEMENT

This study was supported by the Research Program (the specialized research center for future ground system analysis (Contract No. 311165-911074201)) funded by the Agency for defense development in South Korea.

REFERENCES


Macfadzean, R. H. 1992. Surface-Based Air Defense System Analysis, Artech House, Boston, USA.


AUTHOR BIOGRAPHY

HWAN IL KANG was born in Incheon, South Korea and graduated from the Seoul National University, Korea Advanced Institute of Science & Technology and University of Wisconsin-Madison in USA where he studied electrical and computer engineering and obtained his BS, MS and Ph. D. degrees in 1980, 1982 and 1990, respectively. He has been a professor at dept. of Information & Communications eng. in Myongji University, South Korea since 1996. His research interest includes modelling & simulation, military sciences and control applications. His e-mail address is : hwanilkang@naver.com and his Web-page can be found at http://www.mju.ac.kr/~hwan.
Analysis of Design Parameters Based on Bullet Trajectories and Hit Probability

Hwan Il Kang
Sang Yeong Choi
Hyun Soo Kim
Department of {Information & Communications Eng.}^{+}
Mechanical Eng.
Myongji University
Yongin, South Korea
E-mail: hwaniikang@naver.com, metayoung@gmail.com^{+}
saint7777@naver.com^{+}

KEYWORDS
Hit probability, Error budgets, Muzzle velocity, Experimental approach, Analytic Method, Allowable regions, Gun pitch angle, Standard deviation, Combat Vehicle.

ABSTRACT
The parameters related to hit probability for optimal design of combat vehicles affects the performance of fire power. In this paper, we present the procedure how to obtain maximal allowable ranges of design parameters so that the desired hit probability may be achieved. To satisfy the desired hit probability, the design parameters should be confined in some regions. For simulation, assume that two design parameters, muzzle velocity error and gun pitch angle error, play major roles for hit probability prediction. The variation of design parameters results in dispersion of bullets at the target plane. Measuring dispersion at the target is called as an error estimate. We use the experimental approach as the error estimate. Experimental approach uses the unit partial and estimation of total errors. Total errors of muzzle velocity and gun pitch angle are transferred to the total errors at the target through the trajectory equation. The unit partials mean the ratio of the error at the target to that at the fire position. In this paper we use both exact trajectory and approximate trajectory. We are able to obtain the maximal allowable range of errors of both muzzle velocity and gun pitch angle in order to achieve the desired hit probability. To satisfy a given hit probability for the given range, it is shown that the vertical gun pitch angle error and muzzle velocity error should be confined into a first quadrant of an elliptic region.

INTRODUCTION
When we design a four-wheeled or six-wheeled combat vehicle, four important aspects are mobility, fire power, vulnerability and operability of the combat vehicle. Mobility is concerned with how fast it drives on the wild road; Fire power is related to fire performances of arms equipped with combat vehicle; Vulnerability is concerned with robustness against enemy fire powers; Operability refers to human factors of combat vehicle. Among four performances of combat vehicle, the fire power performances plays the most important role in the combat environment. The fire power performance of future ground combat vehicle has three important areas: Error estimates, hit probability prediction, and optimal design framework for hit probability. The fire power performance for combat vehicle with suspension system was studied in the paper (Kang et al. 2015c). Hit probability is a function of error budgets (Weaver 1990; Groves 1963). Error budgets come from errors of subsystems. Errors of subsystem are divided into five categories: errors from internal ballistics, mechanical errors from gun and turret, trajectory errors from external ballistics, meteorological errors of sensors, errors of combat vehicle with road conditions such as gun pitch angle error, gun roll angle error, gun vertical displacement error. Table 1 tabulates subsystems and related errors of combat vehicles (Kang et al 2015b).

Hit probability depends on the errors from subsystems of combat vehicles. Target is located in some distance from the position of combat vehicle. On the target, the dispersion of the bullets are due to the errors from subsystems of combat vehicle. The mean point of impact error (Driels, 2013) is a distance from the aiming point to the mean of the impact points and the precision error is a distance from the mean point of impact to each impact point as shown in Figure 1.

<table>
<thead>
<tr>
<th>Subsystems</th>
<th>Kinds of errors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal ballistics</td>
<td>error of mass of propellant, muzzle velocity error</td>
</tr>
<tr>
<td>gun and turret</td>
<td>aiming error (pitch angle error) mechanical error</td>
</tr>
<tr>
<td>external ballistics</td>
<td>trajectory error</td>
</tr>
<tr>
<td>meteorological sensors</td>
<td>temperature error, density error</td>
</tr>
<tr>
<td>Combat vehicles with road conditions</td>
<td>gun pitch angle error, gun roll error, gun vertical displacement error</td>
</tr>
</tbody>
</table>

In this paper, the estimation of the hit probability is achieved. We estimate errors and predict hit probability. Two selected design parameters among many error budgets is assumed to be a vertical gun pitch angle and muzzle velocity errors. Hit probability and vertical gun pitch angle error was studied (Kang et al. 2015d). The muzzle velocity error is in the category of the interior ballistics while the gun pitch angle error is in the category of combat vehicles with road.
conditions. Then hit probability depends on the vertical gun pitch angle and muzzle velocity errors for a given range.

![Figure 1: Mean point of impact error and precision error](image)

To satisfy the given hit probability for a given range, the gun pitch angle and muzzle velocity errors should be confined into some regions. The maximum allowable region for the gun pitch angle and muzzle velocity errors will be determined.

**ERROR ESTIMATES AND HIT PROBABILITY PREDICTION**

There are two error estimate methods: one is the engineering based method and the other is the experimental based method. These have been described in paper (Kang et al. 2015b). We use the experimental approach as the error estimates. Experimental approach uses the unit partial and estimation of total errors. Total errors of muzzle velocity and gun pitch angle are transferred to the total errors at the target through the trajectory equation. The unit partials mean the ratio of the error at the target to that at the fire position. In addition, there are two methods of hit probability prediction: one is an analytic method and the other is a Monte Carlo method. These have been described in paper (Kang et al. 2015b). In this paper, hit probability prediction may be obtained using an analytic method. The analytic method assumes that hit probability is expressible as the integral of Gaussian distribution of errors at the target.

**TRAJECTORY & RELATED EQUATIONS**

The trajectory differential equations (McCoy, 1999) are written by

\[
\begin{align*}
\dot{v}_x &= -C_d^*v_x(v_x - w_x) \\
\dot{v}_y &= -C_d^*v_y(v_y - w_y) - g \\
\dot{v}_z &= -C_d^*v_z(v_z - w_z)
\end{align*}
\]  

(1)

where \(v_x, v_y, v_z\) are range, upward and deflection velocity, respectively and \(w_x, w_y, w_z\) are range, upward and deflection velocity of winds, respectively (See Figure 2). The constant value \(C_d^*\) is a modified drag coefficient and the value is assumed to be \(1.8253 \times 10^{-4} \text{ m}^{-1}\). The value \(g\) is the gravitational acceleration. The value \(v\) is the scalar value of the velocity vector and is expressible as

\[v = \sqrt{v_x^2 + v_y^2 + v_z^2} \text{ m/sec.}\]

The above standard differential equations are solved by the numerical analysis. On the other hand, assume that the firing angle is very low. Under this condition, we obtain the algebraic solution. Consider the situation that the elevation angle is less than 22.5 degree (Kang et al. 2015a). Under this condition,

\[\left|\frac{v_x}{v_y}\right| \ll 1 \text{ and } \left|\frac{v_y}{v_z}\right| \ll 1.\]

So, the value \(v\) can be described by

\[v = \sqrt{v_x^2 + v_y^2 + v_z^2} = v_x \frac{v_y^2 + v_z^2}{v_y^2 + v_z^2} + 1 \equiv v_x\]

The family of the given differential equations are approximated by

\[
\begin{align*}
\dot{v}_x &= -C_d^*v_x(v_x - w_x) \\
\dot{v}_y &= -C_d^*v_y(v_y - w_y) \\
\dot{v}_z &= -C_d^*v_z(v_z - w_z) - g
\end{align*}
\]

(2)

The first equation in (2) may be solved easily since the differential equation has one variable \(v_x(t)\). Using the explicit expression for \(v_x(t)\), the second and third equations may be solvable. Finally, the algebraic equations can be derived. The solutions are

\[
\begin{align*}
x(t) &= w_x t + \frac{f}{C_d}\n
z(t) &= w_y t - \frac{w_y f}{C_d v_w} \\
y(t) &= -0.25gt^2 + \frac{g - 2v_w w_y C_d^* f}{2v_w^2 C_d^*} + \frac{(-0.5gt + v_w w_y C_d^* t + v_y(0)f)}{V_w C_d^*}
\end{align*}
\]

(3)

where

\[V_w = V_w(0) - w_x\]

\[f = \ln(1 + t C_d V_w)\]
SIMULATION WITH EXACT TRAJECTORY

If we use the differential equations in (1), we need a numerical analysis. One of the numerical method is the Runge-Kutta method. At first, error estimate is taken with exact trajectory. Suppose that the range is set to be \( x = 700 \) m. Assume that the muzzle velocity \( v_{mx} \) is \( 241 \) m/sec and the gun firing angle is \( 5 \) degree. In Table 2, data for height of target and firing elevation angle, nominal range and muzzle velocity are tabulated.

Table 2: Parameters of Target and Nominal Data for Simulation

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Height of target ( b )</td>
<td>2 m</td>
</tr>
<tr>
<td>Initial firing elevation angle</td>
<td>5 degrees</td>
</tr>
<tr>
<td>Nominal range</td>
<td>700 m</td>
</tr>
<tr>
<td>Nominal muzzle velocity</td>
<td>241 m/sec</td>
</tr>
</tbody>
</table>

Then using the trajectory derived from the differential equations, the unit partial \( \frac{\partial y}{\partial \phi_0} \) is obtained by \( 0.3518 \frac{m}{mrad} \) as shown in Figure 4. The unit partial \( \frac{\partial y}{\partial \phi_0} \) means the difference between the vertical displacement at the given range (700m) under the fire angle 5 degree+0.5 miliradian and that under the fire angle 5 degree-0.5 miliradian. Figure 3 shows two trajectories and Figure 4 shows the zoom-in view for Figure 3. As shown in Figure 4, at the target, the variance due to the error of the fire vertical angle is

\[
\sigma_{f,t}^2 = 0.3518^2 \sigma_{\phi_0}^2
\]  

(4)

![Figure 2: Bullet Trajectory (McCoy, 1999)](image)

We also consider a unit partials for muzzle velocity. If we use the differential equation, the unit partial is obtained by \( \frac{\partial y}{\partial v_{mx}} = 0.4522 \) sec as shown in Figure 5. Assuming that \( \sigma_{\phi_0}^2 \) and \( \sigma_{v_{mx}}^2 \) are independent and we use the exact trajectory, the total variance is given by

\[
\sigma_{y,Exact}^2 = 0.3518^2 \sigma_{\phi_0}^2 + 0.4522^2 \sigma_{v_{mx}}^2
\]  

(5)

![Figure 5: Determination of Unit Partial \( \frac{\partial y}{\partial v_{mx}} = 0.4522 \) sec.](image)

SIMULATION WITH APPROXIMATE TRAJECTORY

Instead of using Runge-Kutta method, we use the approximate trajectory in this paper. All parameters are the same as in Table 2. Then using the approximate trajectory derived from the differential equations, the unit partial \( \frac{\partial y}{\partial \phi_0} \) may be obtained by \( 0.3501 \frac{m}{mrad} \) as shown by Figure 6. The unit partial \( \frac{\partial y}{\partial \phi_0} \) means the difference between the vertical displacement at the given range (700m) under the firing elevation angle 5 degree+0.5 miliradian and under the firing elevation angle 5 degree-0.5 miliradian.

![Figure 3: Two trajectories with different gun pitch angle](image)
At the target, the variance due to the error of the fire vertical angle is
\[
\sigma_{\text{Y,approx}}^2 = 0.3501^2 \sigma_{\phi_0}^2
\]
(6)

We also consider a unit partials for muzzle velocity. The unit partial \( \frac{\partial y}{\partial v_{\text{Mz}}} \) means the difference between the vertical displacement at the given range (700m) under the muzzle velocity 241.0 m/sec and under the 241.0 m/sec. shows two approximate trajectories.

If we use the approximate trajectory equations in (2), then the unit partial \( \frac{\partial y}{\partial v_{\text{Mz}}} \) may be obtained by 0.5285[sec] as shown in Figure 7, at the target, the variance due to the error of the muzzle velocity is
\[
\sigma_{\text{Y,approx}}^2 = 0.5285^2 \sigma_{v_{\text{Mz}}}^2
\]
(7)

Assuming that \( \sigma_{\phi_0}^2 \) and \( \sigma_{v_{\text{Mz}}}^2 \) are independent and we use approximate trajectory, the total variance is given by
\[
\sigma_{\text{total,approx}}^2 = 0.3501^2 \sigma_{\phi_0}^2 + 0.5285^2 \sigma_{v_{\text{Mz}}}^2
\]
(8)

**HIT PROBABILITY PREDICTION**

When a bullet hits a target, the dispersion of the bullet forms a Gaussian distribution whose center is a center of the target.

The height and the width of the target are assumed to be \( 2a \), \( 2b \), respectively. Hit probability (MacFadzean 1992) is defined as
\[
p_{\text{ssh}} = \frac{1}{2\pi \sigma_x \sigma_y} \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} \exp \left[ -\frac{(x-h)^2}{2\sigma_x^2} \right] \exp \left[ -\frac{(y-k)^2}{2\sigma_y^2} \right] dx \, dy
\]
(9)

where \( -a \leq x \leq a, -b \leq y \leq b \) and \( \sigma_x \) is a standard deviation along the horizontal axis and \( \sigma_y \) is a standard deviation along the vertical direction. Then, hit probability is expressible using the error function as
\[
p_{\text{ssh}} = \frac{4}{\pi} \int_0^a \int_0^b e^{-t^2} dt \, d \]
(10)

Using the error function
\[
erf(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2} dt
\]
Equation (10) is rewritten as
\[
p_{\text{ssh}} = erf \left[ \frac{a}{\sqrt{2} \sigma_x} \right] erf \left[ \frac{b}{\sqrt{2} \sigma_y} \right]
\]
(11)

Suppose that the height is \( b \) and the vertical standard deviation is \( \sigma_x \) and the horizontal standard deviation is equal to zero, then the single shot hit probability is represented by
\[
p_{\text{ssh}} = erf \left[ \frac{b}{\sqrt{2} \sigma_{x,\text{Exact}}} \right]
\]
(12)

The maximal allowance of the standard deviation of the vertical gun pitch angle is either
\[
\sigma_{x,\text{Exact}} = \frac{b}{\sqrt{2 \text{erf}^{-1}(p_{\text{ssh}})}}
\]
(13)

Using the equation (12), the contour plot is obtained as shown in Figure 8 where \( \sigma_{x,\text{Exact}} \) is given by (5).
If a variable of hit probability is set to be the z axis, we obtain Figure 9. We are in a position to use the approximate hit Probability by Polya-Williams’ approximation (Macfaddzean, 1992). The hit probability is given by

\[ p_{ssh} = \sqrt{1 - \exp \left( -\frac{2b^2}{\pi\sigma_{T,approx}^2} \right)} \]  

(14)

The maximal allowance of the standard deviation of the gun pitch angle and muzzle velocity errors is

\[ \sigma_{T,approx} \equiv \frac{b}{\sqrt{2\ln(1 - p_{ssh}^2)}} \]  

(15)

where total variance is given by (8).

In Figure 10, an allowable region for each hit probability is shown. The region looks like a first quadrant of an elliptic pattern. If hit probability becomes larger, then an allowable regions are smaller. If a variable of hit probability is set to be the z axis, we obtain Figure 11.

Assume that the muzzle velocity error is set to 1m/sec. Under this condition we compare two cases. With exact trajectory, to satisfy the 50% of hit probability, the gun pitch angle error should be less than 1.15 milli-radian while with approximate trajectory, the gun pitch angle error should be less than 1.20 milli-radian.

CONCLUSIONS

In this paper, we have presented the procedure how to obtain maximal allowable ranges of design parameters so that the desired hit probability may be achieved. To satisfy the desired hit probability, the design parameters should be confined in some regions. For simulation assume that two design parameters, muzzle velocity error and gun pitch angle error, play major roles for hit probability prediction. The variation of design parameters results in dispersion of bullets at the target plane. Errors of muzzle velocity and gun pitch angle are transferred to the errors at the target through the trajectory equation. In this paper we have used both exact trajectory and approximate trajectory. To satisfy a given hit probability for the given range, it has been shown that the vertical gun pitch angle error and muzzle velocity error should be confined into a first quadrant of an elliptic region. In the future, we will study optimal design framework with many design parameters larger than two.

ACKNOWLEDGEMENT

This study was supported by the Research Program (the specialized research center for future ground system analysis (Contract No. 311165-911074201)) funded by the Agency for defense development in South Korea.

REFERENCES

Macfadzean, R. H. 1992. Surface-Based Air Defense System Analysis, Artech House, Boston, USA.

AUTHOR BIOGRAPHY

Hwan Il Kang was born in Incheon, South Korea and graduated from the Seoul National University, Korea Advanced Institute of Science & Technology and University of Wisconsin-Madison in USA where he studied electrical and computer engineering and obtained his BS, MS and Ph. D. degrees in 1980, 1982 and 1990, respectively. He has been a professor at dept. of Information & Communications eng. in Myongji University, South Korea since 1996. His research interest includes modelling & simulation, military sciences and control applications. His e-mail address is : hwanilkang@naver.com and his Web-page can be found at http://www.mju.ac.kr/~hwan.
AUTOMOTIVE SIMULATION
Driving Simulator Development
Phase II – Building and Controlling Scenarios through the Motion Algorithm

Alexandra TUCA  Valerian CROITORESCU  Thomas BRANDMEIER
Scientific Research and  Center of Automotive Research on
Continuous Training Center for  Integrated Safety Systems and
Sustainable Automotive Technologies  Measurement Area
University Politehnica of Bucharest  Technische Hochschule Ingolstadt
313 Splaiul Independentei st., 6th Sector, 060042, Room JC004  13 Paradeplatz st., 85049
Bucharest, ROMANIA  Ingolstadt, GERMANY

KEYWORDS
Driving simulator, driver behavior, motion algorithm, control, driving scenarios

ABSTRACT
Driving simulators can be easily and economically configured to simulate a variety of human factors research problems. They allow evaluating and optimizing the human performances within system constraints, being able to indicate the problem areas in system design and functioning. They are particularly useful in selecting a viable system approach from numerous alternatives and evaluating system performances before field testing. Different simulation scenarios can be created to match the requirements of the particular experiment. Vehicle characteristics can be easily modified – the steering ratios, spring rates, damping factors, driven wheels rotary velocity, transmission ratios etc. New roadways or infrastructure can be reproduced using the driving simulator where the test situations are difficult to be reproduced in real life, on the road. They can often represent the most cost-effective approach in a given application. Complete instrumentation and recording systems for in-vehicle tests can be expensive to set up and operate. On the other hand, it is often less expensive to set up and operate simulations in a controlled laboratory environment than it is to conduct field tests that are designed to achieve given experimental objectives. In particular, stimuli and events external to the driver’s vehicle are substantially cheaper to implement, control and vary using a driving simulator comparing with testing on track.

The major disadvantage of research simulators is that the real world will never be replicated in all its complexity. There will always be the issue of validity corresponding to what extent behavior in a simulator corresponds to real life. Although, very powerful computer and graphic systems have been developed lately, the simulation of the real world conditions continues to be problematic.

This paper refers to the problems that can be related both to vehicle dynamics and operation simulation. It completes first phase of the driving simulator development (Tuca et al. 2015) and makes the link to the next needed phases. It will provide a brief overview of the methods which are used to make possible the scenarios development in driving simulators. Building and controlling the scenarios are possible by using dedicated algorithms.

CLASSICAL MOTION DRIVE ALGORITHM

In an ideal world, a simulator would faithfully reproduce the complete range of dynamic cues acting on the driver, from the linear to the angular accelerations experienced during the maneuvering of the vehicle. Furthermore, this would be done in a straight 1:1 manner, such that the acceleration felt in reality would match the one replicated by the simulator. However, a dynamic representation of motion that far exceeds the limited displacement capability of a conventional motion system workspace is needed to simulate typical vehicle handling at this scale demands. In other words, short-lived accelerations at the onset of a maneuver can be reproduced quite accurately, whilst sustained cues cannot. Hence, another technique must be employed to satisfactorily simulate a long-lived acceleration cue. To achieve this, the Motion Drive Algorithm (MDA) filters the vehicle motion before the signals are sent to the motion system.

Only the high-frequency components (onset cue) of the translational and rotational accelerations are reproduced by a corresponding acceleration of the motion system. The low-frequency components (sustained cue) are recreated using tilt co-ordination. The accepted human thresholds of angular motion perception are about 3°/s in terms of angular velocity and 0.3°/s² in terms of angular acceleration (Groen and Bles, 2004).

The classical filter is the most wide-spread (Colombet et al, 2008) of the MDAs (or filters) used today, particularly within the domain of flight simulation. It is most applicable to the range of six-axis motion platforms, known as Stewart platforms or hexapods. These possess six independently actuated legs, where the actuator length can be changed rapidly to vary the platforms’ position and attitude. The Stewart platform allows movement in all six degrees-of-freedom of the Cartesian inertial frame (figure 1):

- surge (forward and backward translation along its x-axis)
- sway (sideways translation along its y-axis)
- heave (vertical translation along its z-axis)
- pitch (tilting rotation around the y-axis)
- roll (tilting rotation around the x-axis)
- yaw (horizontal rotation around the z-axis)
Basic research undertaken at the University of Toronto in the mid-1980s (Reid and Nahon, 1985; Reid and Nahon, 1986a; Reid and Nahon, 1986b) underpins current understanding and utilization of the classical algorithm (Nahon and Reid, 1990).

In the example of driving simulation, the classical filter works primarily on the six orthogonal accelerations generated from the vehicle dynamics model. These are the three linear accelerations of longitudinal acceleration (braking/accelerating), lateral acceleration (cornering) and the vertical acceleration (road roughness and bumps). They are supplemented by the three angular accelerations of pitch (suspension effects of braking/accelerating), roll (suspension effects of handling) and yaw (actual yawing of the vehicle in a turn). To be more accurate, the input to the classical MDA for the linear accelerations is actually the specific force, a description of the linear acceleration with respect to the normal acceleration felt through gravity.

The output of the classical filter describes the desired attitude that the motion platform should adopt, known as the set point. However, in reality, the inertia and mechanical dynamics of the motion platform will delay arrival at the set point. The higher the bandwidth of the motion system is, the smaller the delays are. A typical hexapod bandwidth would be in the order of 5-10Hz.

The horizontal plane specific forces are taken from the vehicle dynamics in the time domain, the signals changing their value continually over the period of the simulation. The main function of the classical filter (figure 2) is to split the time-driven specific forces into the frequency domain, such that their magnitude is described over a range of frequencies that the motion system can realistically achieve.

The low frequency component of the linear acceleration is achieved through a low-pass filter. The corresponding roll of the motion platform is limited to ensure that the tilt (roll in this case) occurs below the perceptual thresholds of 3°/s and 0.3°/s² (Groen and Bles, 2004), in order to “fool” the vestibular system into the perception of sustained lateral acceleration.

The motion system also rotates to mimic the rotational acceleration that the driver would perceive through body pitch roll, determined by the suspension characteristics of the driven vehicle. Like linear acceleration, this rotational acceleration is also high-pass filtered to ensure that its representation exists within the available motion envelope. However, since the acceleration is only short-lived with body roll quickly developing as the vehicle enters the curve, high-pass filtering alone is sufficient as the major components of the existing cueing the high frequency range. The motion platform adopts this roll angle in addition to that commanded by tilt-coordination.

The final element to the classical filter is the ability to reduce the acceleration output that is actually represented by the motion system in relation to the input from the vehicle dynamics model through a scale-factor. Whilst undergoing linear accelerations, humans find estimating the absolute magnitude of those accelerations far more challenging than successfully accessing their relative difference (Bertozzi and Droulez, 1982). By reducing the scale-factor of a particular channel, the classical MDA can be tuned for the worst-case scenario, such that the maximum acceleration to be simulated falls within the motion platform’s displacement limits. However, in such a case only a fraction of the acceleration commanded by the vehicle dynamics model is actually achieved.

In practice, constraints in the design of a hexapod result in interaction of its available modes of motion. For example, significant actuator stroke is required by the demands of pure simulator yaw; this minimizes the available stroke required to achieve demanded excursions in roll or pitch. Hence, in a fully interactive (rather than pre-scripted, such as the case of an entertainment simulator) the simulation engineer is obliged to select even more conservative channel scale-factors for the classical filter.

**PROBLEMS RELATED TO DRIVING SIMULATORS**

Although very powerful computer and graphics systems have been developed lately, the simulation of the real world conditions continues to be problematic. Although problems can be related both to vehicle dynamics and operation simulation, only the last category will be referred here.

**Longitudinal and lateral control of the driving simulator**

It has been generally observed that driving speed is higher in the driving simulator than on the real road. Using moving base simulators, higher speeds were observed on both straight and curved sections of urban and rural roads on the simulator compared to real road (Riemersma et al., 1990). After some experiments it has been observed that the distance to the center line was generally much smaller in the simulator than during driving on the real road and there was a larger variability in lateral position. The same results are
observed for both fixed-base and moving-base simulators, although this problem was attributed to the lack of opposing traffic for the moving-base simulators case.

**Simulation of the road environment and realism**

The simulation of the road environment is very complicated because the more details are provided the more slowly the simulator will run. However, since a certain driving pattern may not be replicated perfectly within a driving condition, the consistency in the driving pattern between different trials within each of the driving conditions (simulator vs. real world) has to be estimated first. In several studies, the question “How realistic do you think the driving in the simulator was?” has been asked. By comparing results from different studies, it is found that the lowest realism and also some “not at all realistic” ratings appear in the studies of anti-collision and vision enhancement systems. These systems are not in common use yet and may add unrealism to the situation. Also, the fact that 5% of the elderly people estimated the driving to be “not at all realistic”, in a study concerning the effects of mobile phone use on elderly drivers may very well reflects that using a mobile phone is an unrealistic task for this section of population.

**Behavior and motivation**

The penalty and reward structure that motivates driver behavior is substantially altered in the simulator. Lives are not at risk. The social and economic pressures that may lead to unsafe driving are also absent. Although monetary penalty/reward schemes can be used to create a motivational basis for behavior using the simulator, it is not clear that this will result in correlation with environment target behavior.

**COLLECTION THE SET OF DATA FROM CAR SIMULATORS**

The measurements were subdivided in two parts: objective measurements and subjective measurements. Outputs from the simulator are included in the objective measurements set. It is possible to record mainly the speed of the car (simulator), the trajectory, deviation from proper lane (to border or to contra-flow-line). These three outputs, combined with reaction time, are basic outputs for analysis of the effect of different physical or mental constraints during driving the car simulator. The simulator enables also to measure movements of pedals (throttle, brake) and movements of the steering wheel. In addition to these simulator outputs, additional devices usage is possible in the simulator or on experimental driver. Outputs from these devices are also included in the objective measurements set.

Subjective measurements are represented for example by the analysis of subjective questionnaires, where the experimental driver describes the own status before measurements, after measurements or during the process of driving the car/simulator. Also the person subjectively evaluates different aspects of tested devices.

**DRIVING PERFORMANCE DATA**

**Speed analysis**

One of the essential factors of a driver’s ability of safe and responsible driving is his/her attention. Attention can be defined as driver’s ability to react promptly and safely to standard and nonstandard situations. Several different experiments were made on the car simulator. The attention of the experimental drivers is purposely decreased by means of the standard activities in the car. One of the factors, which could be easily monitored is the vehicle speed. Probands are instructed to keep predefined speed. During driving the probands are asked to do a certain activity. During this action the driver should split the attention between the task and the driving itself. Due to this fact the driver can lose the correct control and lots of correction actions in his behavior can be found. The correction of the appropriate speed (which is usually lost when performing the given task) represent only one of them. Therefore, demanding tasks cause more variations in the vehicle speed (comparing to the parts when the driver is not disturbed).

**Trajectory analysis (car behavior on the road)**

An analysis of a car trajectory seems to be a very promising and precise classification of driver’s behavior. The classification was done after several differences between the car trajectory and geometrically ideal path were studied. The ideal path is a curve copying the middle of the road that experimenting the person drives on.

**Driver’s reactions**

The driver is permanently in contact with the steering wheel as it is only one control on which the driver keeps hands in standard situation. Therefore the record of the driver is controlling movements of the steering wheel could serve as a basis of very good information of his driving abilities.

**VEHICLE DYNAMICS SIMULATION**

Vehicle dynamics simulation tools, along with their corresponding mathematical computer models, have undeniably become an important part of the vehicle development lifecycle. The quest of numerous vehicle dynamics engineers has led them to modern computerized methods that simplify the creation of complicated vehicle models, as well as the design of highly developed programs for the reconstruction of the physical behavior of individual vehicle systems. A very good correspondence between the real world measurements and the results produced with a vehicle simulation program has been created.

Until recently, however, only the driving situations that can be simulated without a driver (i.e., with time dependent steering wheel angle inputs or at best with simple course controllers) have been examined. Unfortunately, conclusions made from open-loop simulations regarding the closed-loop behavior with a real driver are often difficult, even when conventional vehicles are considered. For modern vehicle concepts (e.g. with four-wheel steering or variable controlled steering ratio) a prediction of the
handling and the mastering of the driver is completely inadequate, since theoretically any dynamic behavior of the vehicle could be realized. The adaptation to the real driver can only be made in the prototype state, which is a very expensive process that is also restricted in the variation and test possibilities.

THI vehicle dynamics simulation: IPG Driver

IPGDriver enables the users to add the control actions of a human driver to your complete vehicle simulations. These actions include the steering, braking, throttle position, gear shifting and clutch operation.

The most important features of IPGDriver are: automatic adaptation to the present vehicle by identifying the dynamic behavior of the vehicle, ability to learn, which gives you the possibility to use the knowledge in later simulations.

Inputs for the driver model include: course on the road, vehicle motions: position, speed and acceleration and steering wheel torque (if available).

Outputs of the driver model are: dependent on the structure of the vehicle model, either the steering wheel angle or the steering wheel torque, standardized accelerator pedal position, force on the brake pedal, position of the clutch pedal, gear number.

Basic concepts

This section describes the general functions of the driver model, particularly the direct setting elements, including: “steering wheel angle”, “position of the accelerator pedal” and the “force on the brake pedal”.

The result of the module “Choice of Course” is a static desired course which will be maintained by the virtual driver. Normally the lane being driven on is much wider than the width of the vehicle, so the driver is free to choose the course between the lane borders. An influence on the desired course can be simulated with different corner cutting coefficients as part of the driver model. Different driver behaviors can be pre-set by the coefficient.

The communication between the basic strategies “Choice of Speed” and “Influence on Speed” (without considering shifting) is shown in Figure 3.

First, the driver has to choose a speed that is suitable for the course. The specified cruising speed represents the speed value until the limit of the lateral speed is exceeded. This “maximum desired lateral speed” characterizes the driver type and can either be a standard value or a specified value - like the cruising speed.

By using the actual speed and the desired speed, the driver can make adjustments such as acceleration or deceleration. When taking action to get the desired acceleration, the driver can rely on the knowledge of the vehicle that was acquired earlier. For example, the driver knows, whether to brake or not, if it is sufficient to simply press the pedal.

The basic strategy for steering is shown in Figure 4. On one hand, the driver determines the course to be driven and may cut corners (basic function „Choice of course”), but adjusts the desired course based on the motion of the vehicle.

Therefore, the desired course is not fixed, it changes according to the vehicle’s motion.

On the other hand, the driver estimates where the vehicle will move. The motion of the vehicle is recognized and the vehicle future course can be predicted. Since the desired path and the predicted path are trajectories, in principle, an infinite number of deviations are possible. However, the driver concentrates on gaining information about the course ahead, within a reasonable distance that is characterized by the preview time. This preview time has an important influence on the ability to control a maneuver. The model will adjust this time to the vehicle dynamics and does not require any input by the user.
If the predicted deviation of the course exceeds a threshold value, the driver will interfere after a reaction time and the steering behavior will be changed. Reaction time and threshold are input values that can change the driver characteristics, such as the level of concentration and experience of the drivers. The sideslip angle and the steering wheel torque are important for the identification of critical driving situations. The self-adaptation to the dynamic behavior of the vehicle, for the speed as well as for the steering strategies, is one of the most important features of IPGDriver. By adapting to the current situation, IPGDriver has made dramatic advancements in the field of driver modelling, and in the process defined itself as the current industry standard. By eliminating the need for large numbers of parameters that must be constantly changed to have an effective simulation, the model can be used for any driver control simulation without the large amount of preliminary work needed for other systems. IPGDriver could be used not only for consideration of the real driver, but also with data input as a course controller. However, this usage, which can be very helpful in daily simulation work, is not going to be described in detail.

The vehicle system

Figure 5 shows a simple diagram of the relationship between the driver, vehicle, and the surrounding environment (which includes the road and the wind) and the forces that act on that vehicle. The arrows show that all three elements interact, and can in fact be thought of as a single system. The driver, the vehicle, the vehicle environment and the interaction (acting forces, input and outputs, etc.) between all of them are called the Vehicle System.

When the vehicle system is modelled in the computer world it can be referred to as a Virtual Vehicle System (VVS). IPGDriver is one component in a virtual vehicle system. The vehicle and environment are modelled with other software (e.g. with IPG’s CarMaker, ADAMS, a MatLab/Simulink model, or some other application).

CONCLUSIONS

In order to build and control the scenarios implemented in the driving simulator, a comprehensive understanding of the basic principle of the motion algorithm was realized. Therefore, the research was conducted to understand the principle and to determine the needed inputs necessary to build and control the scenarios.

Furthermore, the problems related to driving simulator were raised in order to figure out the challenges that will be faced in the future studies and experiments.

The first paper “Phase I – From State of Art to state of work” provided a comprehensive explanation of the driving simulator’s behavior by building the 3D model and analyzing every key sub-system of the simulator.

This paper, the second one from this paper series regarding Driving Simulation, aims to provide understanding on developing and controlling the scenarios, by raising the problems that will be faced in future studies, by understanding the motion algorithm concept and by presenting the IPG Driver. The motion algorithm will allow to comprehend the basic strategies of steering and speed influencing through the vehicle dynamic simulation.

The target is to understand the relationships between driver, vehicle, and simulated environment and to be able to replicate the reality in a driving simulator without any further problems.

REFERENCES


Harlapur, S., “MoCap System”, Technische Hochschule Ingolstadt, Germany, 2014


Modeling and Formal Verification of Reconfigurable Vehicular Platoons

Emma Guerfala
Tunisia Polytechnic School
University Of Carthage, Tunisia
E-mail: emnaguerfala@gmail.com

Anis Koubâa
Prince Sultan University, Saudi Arabia
CISTER/INESC-TEC and ISEP,
Polytechnic Institute of Porto, Porto, Portugal
E-mail: akoubaa@coins-lab.org

Mohamed Khalgui
LISI Lab, INSAT, University de Carthage, Tunisia
SystemsControl Lab, Xidian University, China
E-mail: khalgui.mohamed@gmail.com

Oussama Karoui
Macau University of Science and Technology
Macau, China
E-mail: rtkaroui@gmail.com

KEYWORDS
Platoon, Multi-Agent System, Inter-Vehicle Communication, Reconfiguration, Formal Verification

ABSTRACT

The platoon system is considered as a promising transportation approach that may offer many services such as reducing the fuel consumption or increasing the road capacity. However, with new technologies come new complex issues. Vehicles formation requires a high safety level since it evolves human drivers security. In order to deal with uncertainties in the platoon, this paper proposes to manage the safety issue by a reconfigurable multi-agent architecture handling two different modes: normal and degraded modes when faults occur. Finally, we propose to evaluate the correctness of any platoon by using a formal verification approach. We prove the efficiency of the proposed platoon model for several situations such as merging, following or leaving the platoon by verifying the non-collision safety criterion (i.e.: the distance between vehicles should not reach a reference value).

INTRODUCTION

The increasing number of circulating vehicles in urban areas is compromising the traffic jam and the pollution problems. Many solutions have been proposed to overcome traffic issues such as rapid common transport systems which are not too flexible. An innovative transport system has been under development since 1997 in California named the path project (Shladover 2008). The idea was to make vehicles drive at a very close distance to decrease the air pressure and thus the fuel consumption. This transportation system, also called platoon, is composed of a leading vehicle and a set of following autonomous vehicles driving at a constant distance. This strategy enhances the road safety provided that the leader is a professional driver. Platooning systems require a high safety level to prevent any type of road accidents, and there are many challenges facing its real-world deployment on roads. In the literature, there have been several research works that contributed to the evolution of platooning systems, including, paper (Femandez et al. 2012) which evaluated and investigated the performance limitation of the control model on the safety issue. Another project called Karyon (kernel-based architecture for safety-critical control) in (Casimiro et al. 2013) proposed a system solution for a predictable and safe coordination of autonomous vehicles acting in an uncertain environment. Ostberg and Bengtsson also tackled the safety issue at run-time in (ostberg and Bengtsson 2013) and discussed how to handle the uncertainties by using dynamic safety contracts. One of the major problems is ensuring the safety of the platoon in case of potential failures and malfunctioning of individual platoon components. Several researchers in the literature focused on studying the platoon performance under ideal conditions to improve the control performance, and did not carefully considered the impact of real-life abnormal situation that might affect the platoon safety. Moreover, the involvement of vehicles cooperation through communication network is generating more safety issues to cover. The nature of coordination between vehicles is also an issue to consider. Adopting either a global or a local control has a big influence on the data accuracy. A global control requires a global referential involving the setup of a V2V communication. In the other hand, a local control simply depends on the local perception (El-Zaher et al. 2011). However, environment uncertainties and components failures may occur in the vehicular platoon regardless the coordination approach. Therefore, a platoon model should have the ability to reconfigure its behavior depending on changeable and dynamic parameters of the system. A dynamic reconfiguration consists on modifying the configuration of a system during run-time, contributing to the efficiency of the system (Burns et al. 1997). In this paper, our motivation is to ensure the platoon safety under normal and degraded modes. A normal mode is defined as the operation mode of the platoon when there are no...
failures and all components of the system are behaving as expected. The degraded mode is defined as the operation mode of the platoon when some of its components are not behaving as expected. A robust platoon model should be able to cope with these circumstances. To address this issue, we propose an architecture with two different modes to handle the performance degradation. We categorize failures into two classes according to the reference (Solyom and Coelingh 2013): (1) the hard faults concerning the vehicle components such as sensors, communication nodes, or any other devices and (2) soft faults mainly concerning weather conditions. We suppose at this stage that the degraded mode is triggered by a packet loss in the communication network that we specify according to the (m,k) firm model (Hamdaoui and Ramanathan 1995) (Koubba et al. 2004). A number of packets less superior than m among k successive ones forces the vehicle to switch to the degraded mode. Sensors are then employed to collect the required data. In this paper, we propose to model the platoon using a multi-agent system approach. Agents are autonomous software entities widely used to represent reactive systems. The platoon is composed of two different agents: the leader and the follower. A decentralized control is distributed among platoon vehicles. Therefore, every follower is considered as a local leader. Follower agents, since they are autonomously driven, are executing a control model based on the PID controller (Xavier and Pan 2009) (Wang 2010) in order to obtain the intended global platoons behavior. We also model the joining and splitting maneuvers (Valdés et al. 2006). The joining is the merging of a non-platooning car in the platoon and the splitting maneuver is leaving the platoon. The efficiency of the proposed control system is evaluated through the formal verification approach using UPPAAL model checker. Our model examines the cooperation between the different platoon agents, verifies the joining and splitting maneuvers progress and primarily checks the safety criterion. The contributions of this paper are as follows: (i.) First, the platoon system is designed as a multi-agent system where agents have the opportunity to switch between two modes due to the reconfigurable architecture. (ii.) Second, we propose to study the joining and splitting maneuvers in the platoon system. (iii.) Third, we propose to verify the non-collision safety criterion while the platoon is managing the different maneuvers and switching between the modes by using the formal verification approach. The originality of this work is in proposing a reconfigurable architecture for the platoon system in order to ensure safety in both normal and degraded modes. We also validate the safety of the platoon for joining and splitting maneuvers. The rest of the paper is organized as follows: In section II, we present the related work of vehicular platoon safety. The multi-agent system architecture is presented in section III. Section IV exposes the proposed model of the platoon system. We present the formal verification model in section V and evaluate the results in Section VI. Section VII concludes the paper.

STATE OF ART

In this section, we present the state of the art of platoon systems and try to position our work in relation to previous researches. One of the most important impact factors on the platoon safety is the adopted control model. In a platooning vehicle, a control model depends on two control types. The first is the longitudinal control that maintains the safety distance between vehicles by varying the vehicle velocity. The second is the lateral control which keeps the vehicle on the same trajectory as the next one. Many works proposed control strategies based on the PID controller such as (Ioanou and Ioannou 1994) and (Moskwa and Hedrick 1990). While a physical inspired model is presented by Soo- Jeong Yi and Kil-To Chong in (El-Zahe et al. 2012). Others used the fuzzy logic for the longitudinal control such as, Lee, Toczuk, Jung and Kim (Lee and Toczuk 2003). The cooperation between vehicles is also an important factor. A significant stream of research divided the coordination approach into two main trends on the basis of the information origin (El-Zahe et al. 2011): local and global approaches. The second approach is widely adopted in the platoon solutions due to the good results matching and the need of cooperative entities for some maneuvers and emergency situations as mentioned in (Bergenhiem et al. 2014). On the other hand, platooning vehicles using the local approach only depend on their own perceptions and their low-cost sensors feedback (e.g. no data exchange between vehicles) but this approach may in contrast suffers from the anticipation error (E.-Zahe et al. 2011). Another important platoon characteristic is the computational architecture. Two main models predominate in the literature: the hierarchical architecture is highly present in several works such as (Kolodko and Vlacic 2003). It is generally composed of three levels which are: the vehicle control, the management and the road-vehicle communication levels as described by Michaud in (Michaud et al. 2006). The second type, proposed in both (Sukthankar et al. 1998) and (Arkin 1998), considers the vehicle model as a set of behaviors which offer more flexibility in response to the environment. In order to prove the feasibility and efficiency of the proposed solutions, almost all previous research works conduct experiments either on real robots or on simulators. In both cases only a specific number of scenarios is tested and a very long period is sometimes needed to follow the system evolution and the possible limitations. In (El-Zahe et al. 2012), Contet, Gechter, Gruer and Kouam propose to verify the non-collision criterion using a compositional formal verification (i.e. deducing the safety criterion from the deduction of several properties). However, the presented model is only based on a local approach using the spring-
damper model and no splitting or joining maneuvers are mentioned. From the same authors, the multi-agent system (Contet et al. 2011) was proposed as a model to manage platoon tasks but no data communication is established between the agents. Concerning the failures management, the Karyon project developed a fault model and fault semantics for sensor systems, which allows the abstraction of sensor faults and facilitates their handling for a better perception quality (Casimiro et al. 2013). Although all the related works dealing with platoon systems are interesting, no one considers the failure management. We propose in this paper a multi-agent architecture that manages global and local modes in platoon systems for their safety. A PID is developed too for both modes and new solutions are proposed to handle safety operations of joining-splitting. To guarantee that no collision can occur especially when modes are changed, an exhaustive formal verification is applied by using the UPPAAL environment.

MULTI-AGENT SYSTEM

Under normal conditions, we propose the adoption of a global decentralized approach which decreases the amount of messages that can be transferred all over the platoon depending on vehicle requests. However, a significant amount of data is to be managed by vehicles. Therefore, we propose to consider the platoon system as a multi-agent architecture where each agent is an autonomous entity able of taking its own decisions to meet its design objectives. Our multi-agent system is composed of two types of agents: a leader and a follower. The latter is able of automatically controlling the vehicle. It receives data from other agents by means of communication and perception before executing the implemented control model. The leader, on the other side, is manually controlled by a professional driver therefore the leader agent is basically responsible of managing the platoon as well as determining the platoon properties such as: the inter-distance, the speed, the followed trajectory. The agent architecture, as shown in Figure 1, is inspired from a previous research (Kolodko and Vlacic 2003). It is based on three levels:

- **Communication Unit:**
  On this level, communicated massages are managed depending on the source, destination and request types (e.g. leading, splitting, joining, following). Data are then(i) transferred to the management unit, (ii) sent to other agents (iii) or ignored if the source is unknown. We suppose that follower agents can only identify the preceding, following and leader vehicles, however, the leader has in possession the entire platoon element states in a table. It owns each vehicle details: its destination, its follower, its local leader identities if the car is not at the platoon front and the mode: normal/degraded.

![Figure 1: The Follower Agent Architecture](image)

The mode concept is explained later in this paper.

- **Management Unit:**
  This unit is the decision entity; it is responsible for the agent goal achievement. It receives the information from the communication unit, sensors, HMI and GPS. This unit is composed of two sub units: the first has a main goal of the decision making concerning the mode to activate, the second is running the adequate maneuver algorithm depending on the notification messages. When receiving a message from the previous vehicle, the Checking Sub Unit(CSU) decides which mode to activate depending on a model described later in this paper. The following process algorithm is using the PID controller to compute the reference data which are the vehicle acceleration a and its orientation θ. The unit output is basically the reference data to apply to actuators for the purpose of driving the vehicle to the desired position.

- **Actuators Control Unit:**
  This is the low level control. This unit receives reference data and converts them to applicable variables to steering, throttle and braking actuators.

The multi-agent system is very useful when using the formal verification approach. In fact, it facilitates the vehicles handling and modeling.

PLATOON MODEL

This section describes the proposed platoon model, its characteristics and the adopted control model.
Platoon Characteristics

We propose to develop a platoon model supporting both global and local decentralized approaches. Control is distributed among vehicles in the decentralized approach. Our platoon $P$ is composed of $n$ identical autonomous vehicles $V_i$ with $i \in [1..n]$ and one different vehicle $V_0$ called the global leader (GL); it is driven manually by a professional driver and it is responsible of (i) the splitting/joining maneuvers (ii) the platoon dissolution (iii) defining the platoon characteristics. Each vehicle is defined by its position in a global referential $(X_i, Y_i)$ as shown in Figure 2. A vehicle $V_i$ is considered as a follower of the vehicle $V_{i-1}$ and a local leader (LL) of the vehicle $V_{i+1}$. The decentralized approach may increase the dependency in the platoon since each vehicle is making its own decisions and transferring data to the next car that may be affected by the decision error of preceding leader vehicles unless a powerful control model is set up. In the flip side, it reduces the crucial role of the leader. In the global mode, each two consecutive vehicles are exchanging messages through the V2V communication as shown in Figure 2. Let us suppose a message $m_{i,i+1}$ sent by a local leader $V_i$ to a follower $V_{i+1}$. The message $m_{i,i+1}$ can be either a notification message or a data message. We are interested in this section in data messages since the other type is mainly used for joining and splitting requests. Each $m_{i,i+1}$ must contain the following elements: the local leader orientation $\theta^G_i$ and the local leader GPS coordinates $(X_i, Y_i)$. The vehicle $V_{i+1}$ receives $m_{i,i+1}$, it performs the control process to accomplish the tracking goal. The follower $V_{i+1}$ should gather the following information: the vehicle orientation (its direction to the north) $\theta^G_{i+1}$, the inter vehicle distance between two vehicles $D^G_{i+1}$ and $m_{i,i+1}$ data. A vehicle direction $\theta^G_i$ in the global mode is computed as follows:

$$\theta^G_i = \arctan \left( \frac{X_{N_i}}{Y_{N_i}} \right)$$

with: $X_{N_i}, Y_{N_i}$ vehicle positions obtained by the compass using the vehicle GPS coordinates. The inter vehicle distance in the global mode $D^G$ is computed using the GPS values of both follower and leader (transferred by V2V communication):

$$D^G = \sqrt{(X_i - X_{i-1})^2 + (Y_i - Y_{i-1})^2}$$

In the local mode, a vehicle $V_j$ of the platoon $P$ faces a communication network degradation which leads to the interruption of $m_{i,i+1}$ transferring. In Figure 2, a communication degradation is detected between $V_1$ and $V_2$ therefore the vehicle $V_2$ switches to the degraded mode. Each vehicle $V_i$ is equipped with sensors that we denote $S_i$ to detect: (i) the local leader orientation $\theta^L_i$ (ii) the inter vehicle distance between two vehicles $D^L_i$.

Failure Management

Several factors may influence the normal functioning of the platoon system which causes the degradation of the control performance. Failure cases are already divided into soft and hard faults (Arkin 1998). We can also categorize these failures into (i) local: affect the functioning of one agent or (ii) global: affect the functioning of the entire platoon. This failure organization offers more flexibility to the platoon. For instance, if a vehicle is receiving inaccurate data, only one agent switches to the degraded mode (DM) without deteriorating the whole platoon performance. In this paper, we suppose that the normal mode is actually a global mode (GM) characterized by a good communication quality. In the other hand, the DM is a result of the communication degradation. Thus, it represents the local mode. Since platoon systems are proposed to be functional under any condition, our control model as well as the computational architecture has to handle these failures without deteriorating the system performance. Therefore, we enable the agent to change its behavior depending on the selected mode and consequently obtain differently the
reference data to keep the platoon on a safe and reliable level. We carry out that switching ability by implementing a sub-unit in the management level responsible for the detection of failures or anomalies in the car system. Once the degradation condition is detected, the DM is activated. In order to maintain the good performance of

the platoon in GM, we implement the \((m,k)\) model in the checking sub unit (CSU) to detect the communication deterioration as illustrated in Figure 3. We adopt this model since it responds to our model specifications: the CSU requires a predefined number \(m\) of lost packets among each \(k\) successive packets number. We suppose: \(V_i\) a local leader and \(V_{i+1}\) the follower. When using the \((m,k)\) model we suppose that \(i\) \(m\) is the number of consecutive packets sent by \(V_i\) and received by \(V_{i+1}\), \(i\) \(k\) is a predefined number of successive sent packets. The mode choice for \(V_{i+1}\) is then:

\[
CSU_{output} = \begin{cases} 
DM & \text{if } (m-k,k)_{i,i+1} \\
NM & \text{if } (m,k)_{i,i+1}
\end{cases}
\]

(3)

**PID Control:**

Once the management unit decides which mode to adopt, the tracking algorithm is performed using the PID controller. Each vehicle \(V_i\) is in charge of determining its own reference data (direction and acceleration) using a control model based on the collected information from \(V_{i-1}\) \((i \in [1,n])\). In the platoon system, the tracking depends on two control types: a longitudinal and lateral control.

**Longitudinal Control**

It concerns the braking and throttle actions. To ensure the safety of the platoon, the vertical distance between two successive vehicles should not be lower than the safety distance fixed by the global leader. It is represented by \(d_{ref}\) in Figure 2. For both local and global modes, we measure in each iteration the distance to the preceding vehicle and compare it to the safety distance defined by the global leader in the aim of maintaining the same safety distance. The inter-distance \(D_i^{G,L}\) is bounded for both modes (G:Global, L:Local) by \(d_{ref}\).

In the global control, we use the leader coordinate values to compute the inter-vehicle distance since it provides us with a more accurate PID output. The inter-vehicle distance computed with vehicles coordinates is more precise than distances measured with a sensor due the noise that may be added. Nevertheless, when communication network data are no more fully reliable, we use sensor outputs in local control and change the gain values (used in the PID formula) for a better control result. The controller output is the needed acceleration that leads to the safety distance between vehicles. The PID controller equation applied at time \(t\) for a vehicle \(V_i\) is presented next for both normal and degraded modes.

\[
u_i(t) = K_P^{G,L} e_i(t)^{G,L} + K_I^{G,L} \int e_i(t)^{G,L} dt + K_D^{G,L} \frac{de_i(t)^{G,L}}{dt}
\]

(4)

With: \(i\) \(u_i(t)\): the acceleration value of the vehicle \(V_i\), \(d_{ref}\): the set point which is the reference distance between two vehicles (limit distance), \(K_P^{G,L}, K_I^{G,L}, K_D^{G,L}\): respectively the Proportional, Integrator and Derivative gains constants in both modes, \(e_i(t)^{G,L}\): the error value at time \(t\) is the difference between the measured inter-vehicle distance and reference distance (between \(V_i\) and \(V_{i-1}\)), \(D_i\): process variable which is the current distance between \(V_i\) and \(V_{i-1}\).

**Lateral Control**

It is related to the steering action. Each vehicle \(V_i\) has to apply the same deviation angle as its local leader when it reaches the same position as the latter. As shown in Figure 2, the last vehicle must take the same direction as its preceding vehicle when it reaches \((X_{N1}, Y_{N1})\) position. For both local and global control, we have the same lateral PID since the deviation of the leader can be measured in both cases. This control is intended to position the follower in the same direction as the leader. The vehicle \(V_i\) is changing its direction as follows:

\[
\forall i > 1 \quad \begin{cases} 
For \ t = t_0, \theta_{i-1}^{G,L}(t) = \theta_c \\
For \ t = t_0 + \delta t, \theta_i^{G,L}(t) = PID_{Steer}(\theta_c)
\end{cases}
\]

(5)

With \(\delta t\) is the needed time to travel the inter-vehicle distance between \(V_i\) and \(V_{i-1}\), and \(PID_{Steer}\) is the control function defined later. For this control, the set point is the leader angle at \(t - \delta t\), the error is then the difference between the current follower angle and the set point. The \(PID_{Steer}\) function is as follows:

\[
\theta_{PID} = K_{pa} * e_i(t) + K_{ia} * \int e_i(t) dt + K_{da} * \frac{de_i(t)}{dt}
\]

(6)
With: (i) $e_i(t)$: the error value of vehicle $V_i$, $e_i(t) = \theta_F - \theta_L$, (ii) $\theta_L$: the set point which is the vehicle $V_{i-1}$ direction at $t - \delta t$, (iii) $\theta_F$: process variable which is the vehicle $V_t$ direction at $t$, (iv) $K_{pa}, K_{ia}, K_{da}$: respectively the Proportional, Integral and Derivative Gain constants.

**Joining and Splitting:**

The Global Leader (GL) is responsible for managing the joining/splitting request. The GL manages 3 types of lists: (i) $Pltn.List$: the GL enqueues this list when a joining success notification is received. It contains the identity number $i$ of each vehicle $V_i$ composing the platoon. In the other side, this list is dequeued by GL when it receives a splitting success notification, (ii) $Join.List$: it is a FIFO list and contains identity numbers of vehicles that sent joining requests, (iii) $Split.List$: is a FIFO list and contains identity numbers of vehicles that sent splitting requests. Figure 4 presents the joining process flow chart. We consider a platoon of $k$ vehicles and a vehicle $V_{k+1}$ not part of the platoon. The GL accepts a joining request from $V_{k+1}$ if the platoon fulfills these conditions: the global leader is not treating any request, the $V_{k+1}$ destination $\in$ GL trajectory, both $Join.List$ and $Split.List$ are empty and the number of platooning vehicles performing under the degraded mode respect the $(m,k)$ firm model: we consider $k$ ($k \geq 0$) as the total number of vehicles and $m$ the number of vehicles performing under normal mode. Once the join request is accepted by the GL two cases are presented: If $Pltn.List$ $\neq \emptyset$ // $k > 0$

GL sends its coordinates directly to $V_{k+1}$.
If $Pltn.List = \emptyset$ // $k > 0$
GL assigns the vehicle $V_t$ to communicate with $V_{k+1}$. Consequently, $V_k$ sends $m_{k,k+1}$ to $V_{k+1}$ until the join success.
A splitting request is triggered by the GL. The latter checks constantly follower positions and compares them to their destinations. We suppose $V_t$ is a platoon vehicle that reaches its position. GL sends a split request to $V_t$ before it reaches its destination taking into account the splitting process delay. A split request is always prioritized to other requests unless the $split.List$ already contains split requests. A vehicle can split from the platoon regardless its position (at the rear or in the middle). Figure 5 illustrates the splitting process flow chart. When splitting from the back, the vehicle decreases the speed and changes its direction. Once out of the platoon, it notifies the leader of the split success. When splitting from the bottom, the vehicle does not change its direction until informing its local leader of the new follower identity and informing its follower as well of the identity of the new local leader.

![Joining Process Flow Chart](image)

**Figure 4: Joining Process Flow Chart**

![Splitting Process Flow Chart](image)

**Figure 5: Splitting Process Flow Chart**

**FORMAL MODELING and VERIFICATION**

After presenting the platoon model, we use the formal verification to evaluate the proposed model. We test the different inter-distances, the tracking errors and the impact of splitting and joining in the platoon performance for both global and local modes. The imple-
mented model is based on the platoon system already described. According to the platoon model, the follower is composed of 2 unit and two sub units. Therefore, a single follower agent is represented by 4 automata as illustrated in Figure 6: the follower communication unit automaton, the follower checking unit automaton, the follower management unit automaton, the follower actuator unit automaton. In the other hand, the leader is only represented by one automaton since it is manually driven. One last update automaton is added to update distances and positions values.

Figure 6: Interaction Model Automaton

The leader automaton is responsible for the joining and splitting maneuvers as well as transmitting to its follower needed data. It is able to treat one request at a time otherwise it enqueues requests in the joining or splitting list and treats them following the FIFO principle. The leader automaton has common synchronization only with the follower communication unit automaton. It initiates the platooning process by broadcasting beacons and sends information messages to the follower such as the joining or aborts confirmation as well as the reference data in the global mode. The leader is constantly checking if any follower is near to its destination through the find() function. Once such a vehicle is detected, the leader sends a split request to the involved vehicle if no other communication is taking place otherwise it enqueues the split list. Its automaton model is represented in Figure 7. The follower communication unit is basically in charge of transferring joining and splitting messages from the leader to the checking unit. A communication may be established between two followers either when one of them is joining the platoon as an $i^{th}$ agent or during the following in the global mode as shown in Figure 8. In the first case, the follower itself becomes a local leader and has synchronizations with the communication unit of the second follower. A synchronization example is change.ld to inform the follower of the new leader. The checking unit is responsible for the evaluation of the communication quality. As we can see in Figure 9, the checker is responsible for the switching between the two modes. We model the degradation of quality by a probabilistic variable named Packet Loss in order to switch between both modes. The checking unit sends either normal_mode or degraded_mode signal to the management unit. The management unit consists of three main branches: the joining, following and splitting. In each one of them, the acceleration and the direction of the vehicle are computed after receiving the information from the communication unit. Automaton have been simplified to not overload the paper therefore the PID controller functions are not visible. For each maneuver the PID function is used to compute the reference data to send to the actuator automaton.

Figure 7: The Leader Automaton

Figure 8: The Follower Communication Automaton

Figure 9: The Follower Checker Automaton

Figure 10 shows the transitions where we call
notice that the distance between vehicles remains generally stable as desired. It fluctuates around the 10m value since we defined in the following algorithm that a \( dref = 2m \) is acceptable to avoid the excessive speed variation. These results prove the efficiency of the implemented PID controller as well as the good reception of data through the agent architecture. Figure 14 shows inter-distances while the three following vehicles are performing in the degraded mode which is considered as the worst case. In order to switch to the degraded mode, we use the \((m, k)\) model with \( k = 5 \). We declare a variable PL designing the number of lost packets. For each 5 sent packets we check the PL value. If superior than \( 3(m = 3) \), then we pass to the degraded mode. Since data are collected with a sensor in the local mode, we use the inter-distance equation (3) with an added random errors to implement the inaccuracy of sensors. For each iteration we compute the real inter-vehicle distance and add a random number included in \([-2,2]\). The longitudinal control is clearly less efficient than that for the global control. Nevertheless, the switch to the degraded mode ability ensures the safety property during the following maneuver. We verify as well the smooth progress of splitting and joining maneuvers by evaluating their impact on the non-collision criterion. In Figure 15, \( V_1 \) is the first vehicle joining the platoon. Both \( V_2 \) and \( V_3 \) send joining requests while the leader is already treating the \( V_1 \) request. Once the latter reaches the reference distance(10m), the leader goes to the execution joining list requests with respect to the FIFO priority. For the

![Figure 10: The Follower Management Automaton](image)

the listed algorithms: \( \text{join.pltn}(), \text{follow.pltn}, \text{follow.pltn.degraded}(), \text{split.pltn}() \). Finally, we model the actuators automaton represented in Figure 11 which applies the management unit output to actuators by varying the acceleration and the vehicle direction. The update automaton illustrated in Figure 12 is modeled to increment vehicles positions depending on their speeds using \( \text{increment.pos}() \) function for following agents and \( \text{increment.posL}() \) function for the leader agent. The function \( \text{compute.d}() \) computes the distance between each vehicle and its local leader.

![Figure 11: The Actuators Automaton](image)

![Figure 12: The Update Automaton](image)

**PERFORMANCE EVALUATION**

This section shows the formal verification results made with the UPPAAL software for a 4-vehicle platoon. The leader is supposed to be driven at a constant speed equal to 90km/h. The vehicles are initially initiated with different GPS coordinates and a considerable inter-distance. When executing the simulation, UPPAAL shows a tree exposing the different variables used by automaton such as (i) inter-distance values, (ii) joining and splitting requests identity, (iii) vehicles positions, (iv) each vehicle adopted mode, (v) adopted mode state (joining, following, splitting). We register their different values during the joining, following ans splitting maneuvers. We obtain the different plots illustrated by the following figures. In Figure 13, the three following vehicles are performing in a normal mode. We verify the non-collision property by comparing the different inter-distances with the reference distance set to 10m. We

![Figure 13: Inter Distance in Normal Mode](image)

splitting process the \( V_1 \) is the first to leave. Since \( V_1 \) is the local leader of \( V_2 \), the distance between \( V_2 \) and the leader increases when the \( V_1 \) is performing the splitting. The same thing occurred with \( V_3 \) when \( V_2 \) splits from the platoon. As we can see, inter-distances never fall bellow the reference distance for any type of maneuver which proves the ability of the platoon model to manage different maneuvers and to switch between both modes while ensuring the platoon safety. These results show the validity of our proposed approach. Safety is provided all through the platoon system performance and
for all maneuvers.

Figure 14: Inter Distance in Degraded Mode

Figure 15: Inter Distance within Splitting and Joining Maneuvers

CONCLUSION

In this paper, we propose to consider the platoon as a multi-agent system with a reconfigurable agent architecture enabling the vehicle to perform in two different modes. We validate the application of the platoon model and the ability of the system to switch between different modes without compromising the safety issue using the formal verification approach. In future works, we intend to test our controller performance using a software simulation. We will be also focusing on the management layer and develop a suitable algorithm in order to handle the switching between the platoon modes for an extended and wide range types of failures. Our algorithm may have several inputs such as the weather condition, the road state, the communication quality...etc and a unique output to apply on the vehicle actuators.

REFERENCES


Koubaa A.; Song Y.; and Thomesee J., 2004. Integrating (m, k)-Firm Real-Time Guarantees into the Internet QoS Model. In NETWORKING 2004, Networking Technologies, Services, and Protocols; Performance of


stberg K. and Bengtsson M., 2013. Run time safety analysis for automotive systems in an open and adaptive environment. In ASCoMS®SAFECOMP. HAL.
GRAPHICS
AND
MEDIA NETWORK SIMULATION
VISUALISATION OF PHYSICAL SIMULATION USING JAVASCRIPT PHYSICS ENGINE WITH X3DOM

Olaf Berndt\textsuperscript{1,2} and Linda Huber\textsuperscript{1}

\textsuperscript{1}Fraunhofer Institute for Computer Graphics Research IGD
Joachim-Jungius-Straße 11, 18059 Rostock, Germany
E-mail: \{olaf.berndt, linda.huber\}@igd-r.fraunhofer.de

\textsuperscript{2}University of Rostock
Faculty of Computer Science and Electrical Engineering
Albert-Einstein-Str. 22, 18059 Rostock, Germany

KEYWORDS
X3DOM, JavaScript Physics Engine, Physical Simulation, Modelica

ABSTRACT

In order to visualise the results of a physical simulation and the effect on the related 3D geometry so far only the simulation tools themselves could be applied. A web-based application, though, could provide a powerful alternative for an efficient and realistic visualisation of data enabling users to edit the data and to share their results with customers.

In the course of the project physical modelling and simulation with OpenModelica was coupled with a rigid body simulation via the JavaScript physics engine Ammo.js and an X3DOM-based 3D graphical modelling and visualisation. Now it is possible to start and edit physical simulations as well as to view the effect of changes on the behaviour of the system in the web-based 3D model.

INTRODUCTION

In the scope of the efficient and multiple use of system simulation in the building process of special purpose vessels the connection of data from a physical simulation with a 3D graphical model for the realisation of a 3D Operator Training System is one of the key items (Berndt et al., 2015). Due to the large amount of data and the resulting size of the model it is essential to implement very efficient methods which basically work automatically and rarely need manual editing. In this article we present a method where execution of the physical behaviour simulation including the handover of the variables as well as visualisation of the results of the simulation will be done within a 3D model in the browser. Prior to their combining the graphical and the physical model were created separately. The work was completely realized by the efficient connection of open source tools without individual wrappers or additional frameworks.

As device under test the outrigger of a lifeboat davit system was used. To swing out the outrigger a cylinder is extended until the outrigger is in an upright position. From this position the outrigger continues to move outward due to its inertia and gravity. The functional principle of the davit system is displayed in Figure 1.

![Figure 1: Functional principle of the davit: a) in standby position the davit lies on the cylinder, in case of operation the arm is erected by the cylinder; b) extension of the cylinder stops at the tipping point; c) davit continues moving outward by his own inertia.](image)

RELATED WORK

The web-based visualisation of the results of a physical simulation within 3D models is actually evaluated for different applications. First, there is the possibility to share technical issues obtained from a physical simulation during the engineering process in a visually attractive and realistic environment with the customer (Chu et al., 2015). Furthermore, these methods can also be used for the large field of eLearning (Pang et al., 2013).

GRAPHICAL MODELLING WITH RIGID BODY SIMULATION

X3DOM is used for graphical modelling. It was coupled with a rigid body simulation via the JavaScript physics engine Ammo.js. A comparison with the other physics engines Cannon.js, Bullet.js and JigLib.js showed that Ammo.js fits the key requirements best (Huber, 2013). Beside performance those key requirements were the amount of predefined as well as user-defined shape types and constraints, the possibilities of data exchange and the developing activities. Cannon.js and Ammo.js were nearly equal in overall result, but with individual strengths and weaknesses. Cannon.js convinced with
better performance and ease of operation, but the amount of shapes and constraints was significantly lower. That is why Ammo.js finally was selected.

The original model of the davit corresponds to a real system (Figure 2a) and was provided by a supplier of davit systems. For further use the model was reduced to achieve a better performance, particularly with regard to large 3D objects (Figure 2b). The simplification of the structures, especially for curved and circular objects, resulted in a decreasing number of mesh triangles. In some cases the number of triangles could be reduced by 90%, which resulted in a significant improvement of performance. The reduced model was converted into X3DOM and connected with the physics engine. In this course all objects of the model were converted into rigid bodies. Afterwards these rigid bodies were connected via constraints. The resulting multi-body system out of rigid bodies and constraints can be enhanced by forces and velocities. Thereby drives can be added to the rigid body simulation as well.

![Figure 2](image1.png)

**Figure 2: Model of the Davit; a) Original Model; b) Reduced Model;**

**PHYSICAL MODELLING**

The physical modelling is done with Modelica and the open source tool OpenModelica 1.9 is used as tool for editing and simulating. Regarding the already described test scenario only the cylinder is relevant for the physical modelling as it is the only active component. The physical behaviour model of the cylinder component consists of three parts – the electronic control system, the electro-mechanical drive train and the calculation of the force exerted by the mass acting on the cylinder.

The drive train basically consists of an electric drive and a rotary linear transfer module (Figure 3). There is an electromagnetic force (EMF) which generates a rotation from DC voltage. The EMF is expanded by a rotational inertia, a step-up gear and a brake. The transformation from a rotational to a translational movement is done by a converter. This translational motion moves a mass against a force.

The force, acting against the moved mass, is calculated from the mass of the davit and the lifeboat. The corresponding weight force \( F_M \) is distributed on two force vectors, the force component \( F_C \), which is acting against the cylinder, and the component \( F_B \), which is acting against the bearing at the fixed base point of the davit (Figure 3). The force \( F_C \) acting against the cylinder, depends on the geometry of the davit and changes during the process of erecting the davit. When the cylinder is extended the length \( s \) is raised and the geometry is changed. \( F_C \) is given by:

\[
F_C(s) = F_M \frac{\sin(90°-\alpha)}{\sin(\beta)} ,
\]

with

\[
\alpha(s) = \cos \left( \frac{a^2 + b^2 + c^2 - s^2}{2a(b^2 + c^2)} \right) + \cos \left( \frac{c}{\sqrt{b^2 + c^2}} \right),
\]

\[
\beta(s) = \cos \left( \frac{a^2 + b^2}{{2as}} \right).
\]

![Figure 3](image2.png)

**Figure 3: Physical Model of the Drive Train**

**INTERFACE BETWEEN PHYSICAL AND GRAPHICAL MODELLING**

For the coupling of both components, physical and graphical modelling with rigid body simulation, an interface is required to provide the results of physical simulation for graphical modelling. At first it had to be determined, which data from the physical simulation could be used for the manipulation of graphical modelling. As mentioned before, graphical modelling consisted of a coupling of X3DOM and a JavaScript
physics engine. X3DOM presents the 3D model and can only handle changes in position and rotation as it consists of pure geometric data. The JavaScript physics engine Ammo.js is applied for rigid body simulation and is able to handle physical parameters out of physical modelling. Finally, three parameters are relevant for the coupling: time, velocity and position. Tus, depending on time, it will be determined how fast the inner part of the cylinder will be extended and how far the cylinder itself will be extended. These parameters can be processed by the physics engine in order to calculate the resulting changes in position and rotation of the movable parts of the davit. The results are transferred to the 3D model in X3DOM for visualising the movement of the davit. Graphical modelling is a purely web-based application.

After the coupling of physical and graphical modelling it should be possible to execute all user interactions, e.g. the start of the simulation or modifications of parameters, from the website. While graphical modelling, which is based on JavaScript, completely runs on the client, the application has to be enhanced for the coupling by a Java-based server. This is necessary for editing and executing files of physical modelling and simulation within the scope of a sandbox environment.

After executing OpenModelica the simulation results can be generated as a .csv file. This file can be read by the server after starting the web application or after changing some simulation parameters and will be sent to the client in order to process the simulation results for graphical modelling. Another aim was to control the exported simulation results of OpenModelica via the web application and also to change parameters of the physical simulation, e.g. the combined mass of the davit and the lifeboat, via the web application in order to prevent media discontinuities. OpenModelica saves all parameters and settings for the simulation in an .xml file. Therefore, input fields were added to the GUI of the web application to enter different parameters for physical modelling and simulation, e.g. the mass (Figure 5). After entering the values these will be submitted to the server by pushing a button. The server is responsible for overwriting the corresponding values in the .xml file for the simulation settings of the physical modelling with the submitted values. After that the execution file of the OpenModelica simulation will be started by the server, which will generate a new .csv file with the updated simulation results. If the process of overwriting the .xml file and executing the OpenModelica simulation has been successful the server transmits a positive feedback to the client. Thereupon the client reads the required data for position, velocity and corresponding time stamp from the newly generated .csv file and submits the data to the physics engine for further processing. As a result of rigid body simulation changes in position and rotation of the davit’s movable objects are available and transferred to the 3D model in order to update the visual scene.

CONCLUSION AND OUTLOOK

In the course of the project we realised an efficient web-based coupling of physical modelling and simulation with 3D visualisation using the example of the outrigger of a lifeboat davit system. Changing parameters for the simulation settings and starting the simulation can be executed directly from the website based on a client server architecture. Hence the visualisation of the results of physical simulations within 3D models considering rigid body simulation is realised with open source technology and low demands on hardware and software equipment.

In the next step an interactive Modelica simulation via the web browser is going to be realised. Therefore, the Functional Mock-up Units according to the FMI 2.0 standard shall be connected with a JavaScript-based web application. The use of the Java library javaFMI with this application will be evaluated.

ACKNOWLEDGEMENT

This research has been supported by the German Federal State of Mecklenburg-Western Pomerania and the European Social Fund under grant ESF/IV-BM-B35-0006/12.

REFERENCES


Video replication over DASH in Distributed Multimedia Systems

Haifa Harrouch1, Mourad Kaddes1,2, Claude Duvallet3, Majed Abdouli1,2, Rafik Bouaziz 1
MIR@CL1, Pole Technologique de Sfax BP 242, 3021, Sfax University, Tunisia
FCIT, Jeddah University, Saudi Arabia Kingdom
LITIS3, Le Havre University, 25 rue Philippe Lebon 76600, Le Havre France
e-mail: 1,2firstname.lastname@gmail.com, 3firstname.lastname@univ-lehavre.fr

KEYWORDS
Streaming video, Load balancing, DASH-DMS, Replication, QoS

ABSTRACT

Nowadays, the video traffic represents more than 65% of global traffic and in the Internet forecasts, it will reach 79% in 2018 in both wired and wireless environments. Hence, major content providers such as Netflix, Youtube, Hulu, and Vudu are leveraging HTTP-based multimedia transmission with adaptive streaming with different solution in order to guarantee a quality of service (QoS). Moreover, in 2012, a new standard called Dynamic Adaptive Streaming over HTTP (DASH) which enables adaptation of the media bitrate to varying throughput conditions by offering multiple representations of the same content is proposed. In this paper, we present a fairness architecture, called DASH-DMS, in combination with the new Dynamic Adaptive Streaming over HTTP standard, in order guaranteeing a certain QoS under changing conditions in the available bandwidth. DASH-DMS is a hybrid architecture which combines both 2-tiers and 3-tiers architectures. Moreover, we present a replication policy to enable load balancing video servers and improve the global QoS. Simulations conducted along this paper show that our proposition significantly outperforms existing and state-of-the-art approaches.

Introduction

The recent studies (Stockhammer 2011) have shown the continuous dramatic growth of global Internet traffic. In 1992, global Internet networks carried approximately 100 gigabytes of traffic per day. Ten years later, in 2002, global Internet traffic amounted to 100 gigabytes per second (GBps). In 2012, global Internet traffic reached 12,000 GBps. The Global IP traffic will continue increasing over the next years to reach 50,000 GBps in 2018 (Cisco-System 2015).

In (Stockhammer 2011), the authors have shown that the video traffic dominates the current traffic composition of Internet. The major of this content traffic used the HTTP and benefits from the deployment of existing Internet infrastructure (Stockhammer and Sodagar 2012), (Lederer et al. 2013a). Moreover, to guarantee a QoS and streaming in real-time under varying bandwidth, providers have released adaptive streaming such as Apple HLS, Microsoft Smooth Streaming, Adobe HDS, etc. with limited support of company-independent streaming servers as well as playback clients. These proprietary technologies are not desired. Therefore, standardisation bodies started a harmonisation process, resulting in the ratification of MPEG-DASH in 2012 (ISO-IEC-23009-1 2012), (Lederer et al. 2013b).

MPEG-DASH, also known as DASH, is used with the traditional way of communication client/server that has several drawbacks: (i) It overloads some video servers that are more popular compared to others that are less popular in spite of all video servers (popular or not) containing the same videos requested by clients. (ii) We need to know the address of the video server containing the stream requested by the client in order to view it. (iii) The client is may unable to access the known video server due to failure or maintenance.

In this paper, we propose to implement a DASH in Distributed Multimedia Systems (DASH-DMS). Simulation results show that DASH-DMS improves the success ratio and the quality of video comparing to client/server architecture. Another important issue on VoD is the video replication. The main idea in replication is to keep several copies or replicas of the same resources at different servers. This helps in reducing server load, access latency and network congestion and hence it improves the reliability and availability in distributed systems. However, there are several challenges in deploying such video distribution network e.g., how to efficiently replicate and when to replication access contents so as to achieve the best QoS (Ren et al. 2014). To address these challenges, we propose on-line and off-line algorithms.

The remainder of this paper is structured as follows. In the second section, we present the new architecture DASH-DMS. Then, we propose a replication policy in the third section. The fourth section presents the results of conducted simulations. Finally, in the fifth section, we conclude the paper and discuss some aspects of future work.

Hybrid architecture DASH-DMS

DASH allows video streaming adaptively over the Internet by using the HTTP protocol, which presents an advantage for deployment and provides streaming services for users with dynamic network conditions and heterogeneous de-
vices (Sodagar 2011), (Sanchez et al. 2012) and (Sanchez de la Fuente et al. 2011). However, there seem to be several shortcomings in the use of DASH with basic architecture client/server. Hence, we propose a hybrid architecture DASH-DMS. As its name indicates, it combines the 2-tiers and 3-tiers architectures. Our architecture consists in extending the DASH architecture by adding a primary server used to supervise, control and manipulate our system. As shown in Figure 1, DASH-DMS is made up of:

- A primary server which handles client requests and monitors the status of the system,
- A secondary server which contains a backup of the primary server and takes control when the primary server fails,
- DASH video servers which contain segments and the MPD. They are used to send segments to clients and the MPD to the primary server if necessary. We remind that MPD is a XML file that describes the different versions of the media and their URL.
- Clients make requests to the primary server and receive segments from the DASH video servers.

We will briefly describe a typical procedure executed when a client requests a video under DASH-DMS architecture.

1. The client sends the request to the primary server to retrieve a video.
2. The primary server verifies the existence of the video, its location and the availability of DASH video servers.

Then, it diffuses a request to the DASH video servers available in order to know their status (operational or in breakdown) and recover the MPD.

3. DASH video servers send their responses to the primary server.
4. The primary server sends the merged MPD to the relevant client.
5. Knowing the preference of user and its capacity, the client parses the received MPD. DASH client selects the best suited representation and connects to the DASH video server.
6. The DASH video server sends the segments to the client.
7. The DASH video server sends information to the primary server at the beginning and at the end of the diffusion to perform the updates.

In DASH-DMS, the primary server requests the MPD of video from a DASH video server once. When the file (MPD) is received, it will be stored in database of primary server to prevent is is multiple sending. The content of MPD sending to the clients differs according to the situation:

- If the requested video exist in only one available DASH video server, then the MPD contains the information of this available server.
- If the requested video exists in more than one DASH video server available, then the primary server must generate a single MPD containing the information of the entire available DASH video server organised in order of decreasing availability.

Using the DASH-DMS the load of system is fairly distributed over different video servers. Indeed, only the available video servers that contain the requested video are authorized to answer to client request.

Furthermore, for searching videos, the clients do not handle the addresses of video servers. They have to know only the address of primary server that receives the requests from clients, processes them and sends the answers to clients. When the primary server prepares the answers, it does not take into consideration only the availability of video servers, but also the proximity between the video server and the final user in order to minimise the time of data transfer.

Another DASH-DMS advantage is that it ensures transparency for passage from a video server to another (if possible) when a failure occurs; i.e. when a video server crash. In fact, the streaming of video continues from another server automatically and client continues the reception of the video without knowing.
Replication policy

In VoD network, some video servers can be requested more than others, e.g. some videos are requested more than other one. This may lead to a situation where some video servers may be overloaded while others are under-loaded. Therefore this leads to inefficient use of resources.

To overcome these drawbacks and to enhance the QoS during the overload period, we use a replication policy of videos. Indeed, replication allows a fair distribution of the load among video servers and increases the number of served clients requests. However, there are several challenges in deploying such video distribution network e.g., how to efficiently replicate and when to replication access contents so as to achieve the best QoS. To meet these challenges, we first propose an on-line replication. Then, we consolidate it by an off-line replication.

On-line replication

On-line replication is performed when the primary server finds that only saturated DASH video servers have a copy of requested video. Whereas, the available DASH video servers do not contain the requested video. Hence, it is better to copy a video from a saturated video to available video servers in order to meet client need. However, two problems must be resolved:

- Content replication (CR) problem: since the video exists in different representations and the client will not wait a long time to watch the video, we replicate only the LD representation of the requested video on-line. This is the only representation that can be distributed regardless the debits of clients and also it needs the shorter replication time.

- Server Selection (SS) problem: three types of cases arise
  - If there is one available video server, in this case, it will be the receiver video server (R.V.S).
  - If there is more than a video server available and one of them is near to the client that requested the video, in which case the nearest video server to the client will be the R.V.S.
  - If there is more than a video server available and more than one among them are near to the client that requested the video, in this case the video server that has the greatest bandwidth available among the nearest to the client is the R.V.S.

Off-line replication

On-line replication can temporarily resolve server overload but for a fair distribution of the system load and optimal replication, we propose to enhance our replication policy by: (i) off-line replication policy and (ii) an algorithm for the accomplishment of the replication done on-line.

---

**Table 1: Simulation parameters**

<table>
<thead>
<tr>
<th>Characteristics of the system</th>
</tr>
</thead>
<tbody>
<tr>
<td>Time of simulation</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Characteristics of video</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of videos</td>
</tr>
<tr>
<td>Number of segments per video</td>
</tr>
<tr>
<td>Size of HD video (in KO)</td>
</tr>
<tr>
<td>Size of MD video (in KO)</td>
</tr>
<tr>
<td>Size of LD video (in KO)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Client requests</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrival rates of client demands ($\lambda$)</td>
</tr>
<tr>
<td>Client bitrates (in KO)</td>
</tr>
<tr>
<td>Waiting service</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DASH video server</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of DASH video server</td>
</tr>
<tr>
<td>Number of videos per DASH video server</td>
</tr>
<tr>
<td>Size of bandwidth per DASH video server (in KO)</td>
</tr>
</tbody>
</table>

- Off-line replication: it is done automatically by our system according to the statistics. Periodically, our system must extricate the most requested video. If the video does not exist in enough DASH video servers then the primary server will launch a replication request when system is under loaded.

- Accomplishment replication: it allows completing the replications done online, i.e., to replicate MD and HD representations of the video replicated by online replication.

Simulations and results

To assess the performance of DASH-DMS architecture, we have carried out Monte-Carlo simulations thanks to a simulator developed by our team.

We assume that the client request follows a Poisson process in order to reflect as closely as possible the VoD system in reality. Moreover, in order to obtain significant results, each experiment is repeated 100 times, according to the system parameters given in Table 1. Each value given in the remainder of paper is the average results obtained by 100 iterations.

Influence of DASH-DMS on success ratio

To show the influence of DASH-DMS architecture, we have compared in the first step the results obtained under DASH-DMS with those obtained under basic architecture DASH. In the second step, we have analysed the influence of DASH video server number under the two architectures. Figures 2-4 illustrate graphically this comparison.

Figure 2 shows that when $\lambda$ increases i.e. the number of re-

---

320
quests of videos increases, the success ratio decreases, i.e., the rate of served client requests decreases. However, the success ratio using the DASH-DMS architecture is better than the success ratio under basic architecture regardless of the values of $\lambda$. Moreover, we can deduce from Figure 2 that the number of unfinished requests is most important with basic DASH architecture. The average of failed demands due to the non-existence of requested videos is equal to 35%. However, it does not exceed 5% with DASH-DMS (see Figure 3). This can be attributed firstly to non replication of video in all video servers and secondly to the lack of information about videos, their locations and server video addresses by all clients.

In the following, in order to analyse the influence of DASH video servers number on success ratio, we have varied the number of video servers under basic DASH architecture and under DASH-DMS.

We can see that when the number of video servers increases the success ratio, i.e. clients served, increases regardless of architectures (see Figure 4). Furthermore, under the basic client/servers architecture, the increase of the number of video servers influence rapidly the success ratio principally when the system is moderately loaded. However, under DASH-DMS architecture, the improvement is significant when the system become overloaded, e.g., when the number of video servers increases from 15 to 20 the success ratio increases only when $\lambda$ is greater than 1 (see Figure 4). Hence, we can conclude that using the DASH-DMS permits to distribute fairly the load and optimally uses system capacity. DASH-DMS permits to avoid situations where some servers are overloaded while others are under loaded. Hence, DASH-DMS architecture increases the QoS of global system.

**Influence of Replication**

In previous section, we have shown that when $\lambda < 1$, our architecture: DASH-DMS permits to meet all client requests. However, when $\lambda \geq 1$, we note that the non-existence of requested videos in our system is not the only cause that prevents the processing of customer requests. Hence, in following, we start the simulations from $\lambda = 1$.

**On-line replication**

We can see in Figure 5 that when $\lambda$ increases the success ratio decreases. However, the success ratio using the DASH-DMS architecture with on-line replication is better than the success under DASH-DMS architecture without replication and basic architecture regardless of the values of $\lambda$.

Figure 6 shows that the rates of the low definition (LD) segments are increased for different loads of system; in return the rates of medium and high definition (MD, HD) segments are decreased.

From Figures 5-6, we find that our proposed replication increases the number of customer requests served but it degrades the quality of the segments served. Therefore, we are interested to improve our replication strategy.

**Off-line replication**

Figure 5 shows that when $\lambda$ increases the success ratio decreases. However, the success ratio using the DASH-DMS architecture with on-line and off-line replication is better than the success under DASH-DMS architecture without replication and basic architecture regardless of the values of $\lambda$. However, the success ratio using the DASH-DMS architecture with on-line and off-line replication it is lesser than the success ratio using the DASH-DMS architecture with on-
line replication.

Unlike the on-line replication policy where the success rate of improvement is accompanied by a decrease of quality of delivered video segments compared results obtained with non replication, the combination of offline and on-line replication policies improves the success ratio and maintains the quality of delivered segments near to results obtained with DASH-DMS architecture without replication. This comparison is illustrated by Figures 5-6.

Conclusion and perspectives

The simulations conducted in this study have shown that the use of the new Dynamic Adaptive Streaming over HTTP (DASH) standard under a hybrid architecture (DASH-DMS) improves the global quality of service (QoS) in VoD system. Indeed, we have compare the success ratio of VoD system obtained under classical architecture client/servers and under DASH-DMS and we have shown that DASH-DMS improves the number of requests clients served. We have varied the number of servers in both architectures and we have shown that DASH-DMS offers a better fairness load distribution and increases the overall quality of service.

Furthermore, we have presented two types of replication: on-line and off-line replication. Simulations conducted showed a significant improvement of system load fairness, success rate and quality of video.

In future work, we propose a fault tolerance policy in order to increase the availability of system.

REFERENCES


Lederer S.: Mueller C.: Rainer B.: Timmerer C.; and Hell-


STUDY THE DISTURBANCE SPREAD IN AN INTERACTION NETWORK

Haifa Rabai, Rodolphe Charrier, Cyrille Bertelle
Université du Havre
25 rue Philippe Lebon, Le Havre 76600

KEYWORDS
Interaction network, Chaotic nodes, Coupled Map Network, Source detection, Mean conditional probability of recurrence

ABSTRACT

We study the spread of a disturbance in an interaction network where the nodes have either stable or disturbed state. The arcs represent the interactions. We assume that the interaction network initially contains one single disturbed node that has chaotic time series and may disturb the nodes states in the graph due to interaction. We propose to use a disturbance-propagation model based on a model of non-linear oscillators coupled by an interaction network called Coupled Map Network in order to generate the nodes times series. Taking the inverse problem into consideration, we aim to identify the set of disturbed nodes as well as the source of the spread based on the time series generated by the model. To do that we use some adequate measures from the information theory and we propose a detection algorithm that enables to find the source starting from any disturbed node. We present our simulations to test this algorithm.

INTRODUCTION

To analyze the phenomenon of information diffusion in some real world environment such as internet, social networks or crowds, one can model them as interaction networks where the nodes represent the interacting elements and the arcs are the interactions. On the other hand, the nodes of these graphs have discrete or continuous states. Usually, the diffusion process occurs when the nodes share information about their states via the interaction leading to modify in some cases their dynamics.

Let’s consider for example the epidemiological models where the nodes may have one of the following states: "Suspected", "Infected" or "Recovered" (Christley et al. 2005). Concerning the study of knowledge spread, we typically find two states that are: "Interested" and "Adopted" (Gruhl et al. 2004).

Among the issues related to the problem of information propagation in interaction networks, we are interested in analyzing the propagation of a disturbance. To do this, we consider an interaction network where the disturbance is related to the internal state of the nodes and is reduced to a time-dependent real variable. This state can be "stable" or "disturbed" depending on its dynamical evolution. The interaction between nodes enables them to exchange information about their internal states. Thus, a node with a stable state may become disturbed if it interacts with another disturbed node.

The situation that we study is as follows: we assume that our interaction network initially contains one single disturbed node. First, we aim to detect the nodes that become disturbed due to the interaction in the network. Then, we want to identify the source of this spread among the disturbed nodes.

To tackle this problem, we will proceed in two steps: the first one corresponds to the "direct problem", that is we will make a model of the system and its mechanisms and the second one corresponds to the "inverse problem", that is we only have experimental data to study the problem, except that our "experimental data" here are generated by our preceding first step.

To be more precise, first of all, we model our graph as a specific type of interaction networks that is known as Coupled Map Network (CMN) (Shibata and Kaneko 2003). This model enables us to generate the nodes states as potentially chaotic time series. This means that in our propagation model we know the nodes states and we can guess the set of nodes that may be impacted by the spread of the disturbance.

In the second phase of our analytical approach, we study the inverse problem. In this case, we ignore the network’s structure and we need to analyze the times series generated in the first phase. Therefore, given a snapshot of the diffusion process and based on only the nodes time series, we aim to detect the set of nodes impacted by the spread of the disturbance and to identify the source of this propagation in the interaction network.

Our recent works using Shannon Entropy showed that it is an efficient measure to detect chaotic time series (Rabai et al. 2014).

Here, we will focus more on the second problem which is the identification of the source. A few approaches on this line of work have been proposed in the literature. Shah et al. (Shah and Zaman 2011) tried to find the source of the spread of rumors in a variant of the Susceptible-Infected-Recovered (SIR) model using a maximum likelihood estimator. The graphs that they
studied contained one single infection node. The infected nodes could never recover. The only data that they had was the infected sub-graph. Zhu et al. (Zhu and Ying 2012) studied the problem of detecting an information source in a SIR model using a sample path based approach. The source is considered then as the node with minimum infection eccentricity that is the maximum distance between this node and the infected nodes.

In this paper, we put our focus more on investigating the nodes times series as we don’t know the network’s structure. To do this, we use appropriate measures from the information theory. We propose a search algorithm based on the mean conditional probability of recurrence (Romano et al. 2007) that we compute on the close returns plots (Mindlin and Gilmore 1992). The algorithm enables us to find the source starting from any disturbed node. The results depend on different parameters. We will study the impact of these parameters on the detection of the source.

This paper is organized as follows. First of all, we present our disturbance-propagation model that is the Coupled Map Network that generates the data set. Then in the following section, we present the inverse problem by starting with an overview of the measures that we intend to use. We also present the algorithm, our simulations and the results obtained for the algorithm. To conclude, we summarize the main ideas presented in this paper and present some perspectives.

THE DISTURBANCE-PROPAGATION MODEL

The disturbance-propagation network that we study is a dynamic and directed graph $G = \{X, E\}$:

- The set of nodes $X = \{X_1, X_2, ..., X_n\}$ is characterized by measurable state variables $X^t = (x_1^t, x_2^t, ..., x_n^t)$ that may exhibit chaotic time series.
- The set of edges $E = \{E_1, E_2, ..., E_n\}$ defines the interactions between the nodes.

Such graph representation leads us to use a specific kind of interaction networks known as Coupled Map Network (Koiller and Young 2010). The latter is a dynamic system with a discrete time step and continuous states space. The nodes of this model have a measurable state variable that may be chaotic. They are distributed in space and interact with each other by coupling.

We use this model to generate the nodes time series. This is done as follows. We use a non-linear application which is the logistic map (Phatak and Rao 1995) that enables via its control parameter to generate either "disturbed" or "stable" state. A state $x_k$ of a node $X_k$ that is generated by the logistic map, is given by:

$$x_k^{t+1} = f(x_k^t, a_k) = 4 \cdot a_k \cdot x_k^t \cdot (1 - x_k^t) \quad (1)$$

where $a$ is the control parameter, its value is between 0 and 1. According to the bifurcation diagram (cf. figure ??), if $a$ is equal to 1, we have a disturbed state, that is a chaotic time series which is characterized by a positive Lyapunov exponent (Rosenstein et al. 1993). On the other hand, if this parameter is under about 0.86, we have a stable state that may be either periodic or a fixed point behavior. Figure 2 shows some examples of these behaviors.

The dynamic of a node resultant from the interaction with the other nodes is given by the following equation:

$$x_k^{t+1} = (1 - \epsilon) f(x_k^t, a_k) + \epsilon \frac{N_i}{N} \sum_{l=1}^{N_i} f(x_l^t, a_l) \quad (2)$$

where $\epsilon$ is the coupling coefficient that quantifies the strength of the interaction between nodes. It has continues values between 0 and 1. $N_i$ represents the set of nodes directly connected to the node $X_k$ with whom node $X_k$ interacts.
The interaction between nodes is modeled by a coupling process which can propagate a disturbance in the network and generates a partial synchronization phenomenon. Synchronization is defined as a correlation between time series meaning that two behaviors initially different may become identical. This correlation depends on the parameter $\epsilon$. This coupling strength determines the amount of information shared by the nodes. The nodes that are directly connected to the source receive a portion of its state and become disturbed after a sufficiently long time. On the other hand, the nodes that are indirectly connected to the source receive the chaotic information through the disturbed nodes. They can become disturbed in their turn if the nodes with whom they interact are sufficiently affected by the chaos. Therefore, the chaotic information transmitted by nodes decreases as one gets farther from the source. This means that the nodes that are far away from the source are less impacted by the spread of the disturbance.

We will explain the effect of the parameter $\epsilon$ on the nodes states based on the figure 2. When we have a weak coupling, a node may not be impacted by the disturbance and thus have a stable state (cf. figure 2(a)). When we increase the coupling strength, the time series becomes disturbed like in the example 2(b). Finally, when the coupling is strong enough, the node’s state becomes more impacted by the chaos as if it has been generated with $\alpha$ equal to 1 (cf. figure 2(c)) like we have done for the source.

In the CMN model, we know the source and we can guess the set of nodes that may be impacted by the spread of chaos. However, we need measures to confirm our assumptions. In the next section, we consider the inverse problem and we present these measures.

**ANALYZING THE INVERSE PROBLEM**

**Detecting the set of disturbed nodes**

To characterize the dynamic properties of the node time series, we need a measure that can detect the chaotic nature of a signal. One can think of using the Lyapunov exponent (Rosenstein et al. 1993) to achieve this goal. However, it is difficult to compute it on any kind of interaction network and on experimental data (Wolf et al. 1985). Therefore, we propose to use the recurrence quantification analysis and more precisely the Shannon entropy that we compute on the close returns plots (Mindlin and Gilmore 1992).

Close returns plot is a binary matrix $N \times N$ that has been proposed to analyze and visualize the behavior of non-linear dynamical systems. Let $x_k$ be the time series of a given node $X_k$. The principle consists in comparing this time series at each time step $i$ to itself with a delay $j$.

\[
R_{X_k}(i,j) = \theta(\delta - |x_k^i - x_k^{i+j}|) \tag{3}
\]

where $\theta(x)$ is the Heaviside function: if $\delta - |x_k^i - x_k^{i+j}| > 0$ then $\theta = 1$, otherwise, $\theta = 0$.

If the difference $x_k^i - x_k^{i+j}$ is under a certain threshold $\delta$, $x_k^i$ and $x_k^{i+j}$ are said to be recurrent and are represented by 1 in the matrix. Otherwise, they are considered non-recurrent and 0 is put in $(i, j)$. Figure 3 shows an example of close return plot. The black segments of this plot correspond to recurrent segments, the white ones are non-recurrent segments.

The threshold $\delta$ enables to discretize the data and determines the structure of the close return plots. It is calculated as follows:

\[
\delta = (\min_{k=1..n}(x_k^i) - \max_{k=1..n}(x_k^i)) \alpha \tag{4}
\]

where $\min$ and $\max$ are respectively the minimum and maximum values computed over all the values of the nodes time series. Different rules Schinkel et al. (2008) have been proposed to compute the threshold $\delta$, the most used one consists of choosing a parameter $\alpha$ that is equal to 10%. The values of the time series of our nodes ranging between 0 and 1, we define a global threshold for all nodes. $\delta$ will have the following values for our study: 0.1, 0.01 and 0.001.

Many measures were proposed later by Trulla et al. (Trulla et al. 1996) to transform the graphical interpretations into statistical analysis. Among these measures were the recurrence rate and the Shannon entropy that is correlated with the inverse of the largest Lyapunov exponent.

The recurrence rate is defined as the sum of recurrent points in the close returns plots divided by the total number of elements in the matrix:

\[
RR_{X_k} = \frac{1}{T^2} \sum_{i=1}^{T} \sum_{j=1}^{T} R_{X_k}(i,j) \tag{5}
\]

The Shannon entropy is given by:

\[
S = -\sum_{g=1}^{n} P_g \log(P_g) \tag{6}
\]

where $P_g$ is the probability to observe a non-recurrent segment with length $g$, that is the number of non-recurrent horizontal segments with length $g > 0$ divided by the total number of non-recurrent segments (Rabarinmanantsoa et al. 2007). $H$ is the longest sequence of recurrent elements.

The entropy is positive when the node’s state is disturbed, it grows gradually as the time series is affected by chaos.
Identifying the source of the disturbance

The Shannon entropy enables to identify the set of disturbed nodes. We use this set to find the source of the disturbance starting from any disturbed node. Note that we only have time series and we ignore the interactions. Therefore, in order to detect the source, we must be able to quantify the chaotic information transmitted by the nodes. To achieve that, we use information theoretic measure that is the mean conditional probability of recurrence.

The mean conditional probability of recurrence (MCR) is a measure that was proposed by Kurths et al. (Romano et al. 2007) to infer coupling directions between 3 dynamical systems. It is computed based on the close returns plots and their extension to joint recurrence plots. The computation of this measure is done as follows. The first step consists in building a close returns plot for each node of the interaction network. Then, for each couple of nodes $X_i$ and $X_k$, we perform a Hadamart product between their close returns plots which gives us the joint recurrence plots:

$$ JR_{X_iX_k}(i,j) = R_{X_k}(i,j) \otimes R_{X_i}(i,j) $$  \hspace{1cm} (7)

The final step consists in computing the mean conditional probability of recurrence (MCR) that is defined as follows:

$$ MCR(X_i|X_k) = \frac{1}{T} \sum_{j=1}^{T} p(x_i^j|x_k^j) $$

$$ = \frac{1}{T} \sum_{j=1}^{T} \sum_{t=1}^{T} \frac{JR_{X_iX_k}(i,j)}{\sum_{t=1}^{T} R_{X_k}(i,j)} $$  \hspace{1cm} (8)

where $p(x_i^j|x_k^j)$ is the probability that the trajectory of $X_i$ recurs to the neighborhood of $x_i^j$ under the condition that the trajectory of $X_k$ recurs to the neighborhood of $x_k^j$.

If $X_k$ drives $X_i$, we have $MCR(X_i|X_k) < MCR(X_k|X_i)$. Otherwise, if the two nodes are independent we have:

$$ MCR(X_i|X_k) = \frac{1}{T} \sum_{j=1}^{N} p(x_i^j|x_k^j) $$

$$ = p(x_i^j) $$

$$ = RR_{X_i} $$  \hspace{1cm} (9)

where $RR_{X_i}$ denotes the recurrence rate of the node $X_i$. In our research work, we compute the MCR for more than 3 nodes that inherit their behavior from the same node. The increase of the coupling parameter leads to a decrease of the nodes recurrence rate. We assume then that the nodes that are more impacted by chaos have a lower recurrence rate. Therefore, we don’t expect to have necessarily the same inequality of the mean conditional probabilities of recurrence between the driver and the response as proposed by Kurths et al.. Suppose $X_k$ is the driver and $X_i$ is the response, we may have $RR_{X_k} < RR_{X_i}$ and therefore, $MCR(X_k|X_i) < MCR(X_i|X_k)$ (Rabai et al. 2014).

1. Choose an initial node $X_i$ among the chaotic nodes $N_c$.

2. Search the lowest MCR between this node and all the other chaotic nodes.

3. This MCR gives us the next node $X_k$ to be processed. This node influences the most the node $X_i$. Return by starting from this node to step 2.

4. The previous loop ends when we don’t find a node that influences $X_i$. Therefore, this node is considered as the source $X_s$ of the chaos spread.

The MCR algorithm depends on two parameters that are: $\epsilon$ that is the strength of the interaction between nodes and $\delta$ that represents the threshold of the construction of the close returns plots. We will study the influence of these parameters on the success rate of our detection algorithm.

SIMULATIONS

We simulated 100 random graphs that are formed by a number of nodes which varies from 50 to 200 nodes. Here we are not concerned with the impact of the size of the graph on the spread. Therefore, we chose to do our study on graphs of smaller sizes to ensure there is enough nodes affected by the disturbance. Initially, each graph has a disturbed / chaotic node that is likely to impact other nodes that interact directly or indirectly with it. It should be noted that the chaotic node is never influenced by the other nodes.

First, we used the CMN to generate the nodes time series. In this model, we know the nodes states as well as the structure of graphs. In the inverse problem however, we seek to build the
CMN only based on the nodes time series as we don’t know the interactions between them.
To test the detection algorithm, we will study the impact of the parameters \( \epsilon \) and \( \delta \).
The interaction strength \( \epsilon \) is assumed to influence the extent of the spread. Moreover, a strong coupling leads to a synchronization phenomenon. To confirm these assumptions, we varied the coupling strength \( \epsilon \) in our simulations from 0.05 to 1.
With regard to \( \delta \), this threshold determines the structure of the close returns plots. The larger the threshold is, the greater the matrices contain recurrent points. Therefore, it is interesting to study the influence of different values of \( \delta \) on the source detection. Therefore, we have defined thresholds with the following values: \( \delta_1 = 0.1 \), \( \delta_2 = 0.01 \) and \( \delta_3 = 0.001 \).
Finally, we summarize our study plan for each studied graph. We replayed 3 times the simulation by changing at each time the threshold \( \delta \). For each simulation, we assigned to the coupling strength 20 values ranging from 0.05 to 1. For each coupling value, we selected randomly five starting nodes to look for the source. So basically, for each graph we have achieved 300 searches.

**Results**

To detect chaotic nodes, we built the close returns plots related to each node by using different thresholds \( \delta \). Then we computed the Shannon entropy (cf. equation (6)). Nodes that have positive Shannon entropy are considered as disturbed. It should be noted that the number of impacted nodes grows when we increase the coupling strength \( \epsilon \).
We present in figure 5 the Shannon entropy values for the following configuration: \( \delta = 0.1 \) and \( \epsilon = 0.85 \). Chaotic nodes are colored in red. This result matches the CMN configuration.

**Effect of \( \delta \) and \( \epsilon \) on the source detection**

To identify the source of the spread of chaos among the set of chaotic nodes, we computed the mean conditional probability of recurrence. For each graph, we varied the value of \( \delta \) and the value of the coupling strength from 0.05 to 1. We report our algorithm success rates in figure 4.
Figure 4 shows that when the threshold is too large, we have many errors of detection. In fact, for \( \delta_3 \), we have an error rate that is greater than 50%. These results are confirmed by table 1 where we have the average error percentages \( E_1 \) according to the threshold \( \delta \) computed for all the values of epsilon and for the 100 graphs.
These results are explained by the fact that when we expand the threshold, meaning that we decrease the value of \( \delta \), we obtain more recurrent segments in the close returns plots. However, we aim to detect the influence of chaotic nodes which corresponds to non-recurrent segments.

With regard to \( \epsilon \), when we have a strong coupling (\( \epsilon > 0.9 \)), we can’t find the source any more. We will try to explain this failure.
First, we examined the nodes positions compared to the source. In the majority of cases, these nodes are directly connected to the source. To understand this phenomenon, we compared the recurrence rates (cf. equation (5)) of the chaotic node and an other disturbed node identified as source by our algorithm.
Figure 5 shows the evolution of the recurrence rates according to the coupling parameter, computed on close returns plots built with a global threshold equal to 0.1. When the coupling is strong, the recurrence rates become almost identical. We can conclude that this is due to a synchronization phenomenon that enables to two different states initially different to become almost identical.

![Figure 5: Shannon entropy values. The nodes impacted by the spread of chaos have a positive Shannon entropy and are colored in red.](image)

<table>
<thead>
<tr>
<th>( \delta )</th>
<th>( \delta_1 )</th>
<th>( \delta_2 )</th>
<th>( \delta_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( E_1 )</td>
<td>4</td>
<td>9</td>
<td>56</td>
</tr>
</tbody>
</table>

Table 1: Error percentages according to the threshold \( \delta \).

**Impact of the distance on the source detection**

The nodes mistakenly detected as source are mostly in direct interaction with the chaotic node. We seek to
Figure 4: Success percentages for the global thresholds $\delta_1$, $\delta_2$ and $\delta_3$. When we decrease the value of the threshold ($\delta_3 = 0.01$), we have more detection errors.

Figure 6: Recurrence rates of the source and an other disturbed node according to different values of $\epsilon$. When the threshold is superior to 0.9, the recurrence rates become almost identical.

understand the impact of the positions of these nodes on the error rates of our algorithm. The error rate $E_2$ becomes proportional to the distance and is calculated as follows:

$$E_2 = \frac{1}{D} \sum_{k=1}^{M} d_k$$

where $d_k$ is the distance between a node $X_k$ and the chaotic node, $X_k$ is a node being mistakenly identified as the source. $M$ is the set of nodes mistakenly identified as sources by the algorithm. $D$ is the maximum distance between a disturbed node and the source, it is given by:

$$D = \max \{d(X_s, k) | k \in N_c\}$$

where $X_s$ is the source and $N_c$ represents the set of disturbed nodes. We report the average error percentages computed for our 100 graphs in table 2. We compared these results to those obtained in table 1. When we take into account the nodes positions, the error rate decreases considerably.

We studied the impact of the maximum distance $D$ on the detection of the source in the interaction network. We report in figure 7 the proportional error percentages according to the distance $D$ for our algorithm. The errors related to the source detection increase when $D$ is small. Also, when the extent of the the spread is important, there are less detection errors.

**Discussion**

We used the Shannon entropy computed on the close returns plots to detect the nodes chaotic time series and we succeeded in identifying the set of nodes impacted by the spread of the disturbance.

With regard to the source of chaos, we first studied the impact of the threshold $\delta$ and the coupling coefficient $\epsilon$. Based on the results obtained, we can conclude that our algorithm has the best success rates when we have a threshold $\delta$ equal to 0.1. Moreover, to avoid the errors related to synchronization phenomenon, we must assign to the interaction force $\epsilon$ a value less than $0.9$. Finally, we studied the impact of the nodes positions identified by mistake as sources on the source detection. Each node has a weight in the computation of the er-

<table>
<thead>
<tr>
<th>$\delta$</th>
<th>$\delta_1$</th>
<th>$\delta_2$</th>
<th>$\delta_3$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$E_2$</td>
<td>0.9</td>
<td>2.2</td>
<td>12.9</td>
</tr>
</tbody>
</table>

Table 2: Error percentages proportional to the distance.
ror rate that depends on the distance from the source. Nodes that are directly connected to the source have a lower weight than nodes that are away from the source. When we take into account the position of the nodes identified by mistake in the calculation of the error rate, it drops considerably. Furthermore, the distribution of error rates according to the distance D (cf. equation (11)), meaning the size of the sub-graph affected by chaos, shows that when the extent of the spread is large, we have fewer detection errors.

Therefore, we can conclude that the detection errors are due essentially to nodes that are directly connected to the chaotic node and that we confuse with the source.

PURPOSE OF THE STUDY

We plan to apply our disturbance-propagation model on the problem of stress or anxiety spread in emergency situations.

Psychiatrists and neurologists were interested in anxiety and stress by studying the dynamics of heart rate and respiratory frequency of a group of patients suffering from panic disorder. They concluded that these dynamics were chaotic due to anxiety Caldirola et al. (2004). On the other hand, Fast et al. Fast et al. (2015) considered anxiety as an emotional contagion that spreads from one individual to another within a social network. Based on these studies, we can assimilate the anxiety to the chaotic physiological data that we can obtain via sensors. The disturbed state of the nodes would therefore correspond to physiological data whose dynamics are chaotic.

Similarly, our state variable could match the nervousness parameter of Helbing social forces model Helbing et al. (2002). This particular model takes into account emergency situations in which the transition from “rational” behavior to “irrational” behavior is managed by a single parameter called nervousness. This parameter influences the motion decisions of the pedestrians.

CONCLUSION

In this paper, we presented our work which consists in studying the propagation of a disturbance in an interaction network based on a propagation model called CMN.

The Shannon entropy, computed on the close returns plots, has been an efficient measure for detecting the disturbed nodes.

We also studied the detection of the source of the disturbance in the network by proposing an algorithm that is based on the mean conditional probability of recurrence. Simulations enabled us to test our algorithm and to find the best values of its parameters in order to minimize the error rates.

For future work, we aim to study the effect of the position of the source and the topologies on the propagation of the disturbance in networks.

REFERENCES


VIRTUAL REALITY SIMULATION
3D SURGERY SIMULATION SOFTWARE DEVELOPMENT KIT

Anton Ivaschenko
Samara State Aerospace University
34 Moskovskoye shosse, 443086,
Samara, Russia
E-mail: anton.ivashenko@gmail.com

Alexandr Kolsanov, Aikush Nazaryan
Samara State Medical University
89 Chapayevskaya st., 443099
Samara, Russia
Email: avkolsanov@mail.ru

Andrey Kuzmin
Penza State University
40 Krasnaya street, 440026
Penza, Russia
E-mail: flickerlight@inbox.ru

KEYWORDS
Surgery training, simulation, surgery skills, 3D modeling, software development kit.

ABSTRACT
This paper describes a complex solution for medical surgery education training based on an extensive use of simulation technologies. Based on the experience of “Virtual Surgeon” and “Inbody Anatomy” projects aimed to develop specialized training suites for endovascular and laparoscopy surgery and anatomy training at medical universities there was developed a set of 3D models, simulation scenes and scenarios and software components that can be autonomously used by themselves in future projects. Specially designed Software Development Kit (SDK) is introduced to provide IT developers a platform to build new simulation technologies for medicine. The results are illustrated by the simulation center established at Samara State Medical University

INTRODUCTION
Simulation training is one of the most promising areas of active media technologies application. An example of it is an intensive adoption of simulation technologies in surgery higher education, which requires studying of both deep theoretical knowledge and fail-safe practical skills. Despite a variety of solutions presented at the market, e.g. for laparoscopy and endovascular surgery training, there is a number of challenges of their application in practice. Firstly, there is no universal procedure of their integration into educational process. Secondly, the methodology of surgery training differs from one university to another. Thirdly, most of the existing training suites require tutor’s assistance and cannot adapt to student’s personality and consider the human factor.

These challenges can be solved by implementing the most up-to-date information technologies, advanced visualization and artificial intelligence. Simulation should become attractive for students to encourage them to obtain new skills, at the same time the training suite should understand the gaps and weaknesses of the student individually and adapt the training procedure to provide stable and long term effect.

First results in this area were achieved under the “Virtual Surgeon” and “Inbody Anatomy” projects that were carried out in 2012 – 2014 and include a number of simulation training suites for endovascular and laparoscopy surgery (Ivaschenko 2013, 2014). These products were successfully probated in several medical universities. During the process of their deployment and technical support there was gathered a remarkable feedback from professors and IT departments.

There was identified a strong request to separate the training suites into a number of autonomous components that can be used by universities themselves to build own solutions and adapt them for specific educational programs.

This feedback led to motivation of a new research project started in 2014 and aimed to develop a number of software components for surgery training, an extensive set of 3D models of human body, and special software that can be used by universities to develop their own products capable to provide personalized simulation and training. In addition to it there was designed an Internet platform that provides to its users an opportunity to exchange and share new simulation technologies and products in integrated information space.

To meet these goals there has been developed a specialized software development kit (SDK), which contains a number of components that can be used to implement a large variety of simulation solutions for surgery training. The proposed approach allows the developers of new training suites to better specify the requirements, concretize the scope, prepare effective tests and improve training efficiency. The basic challenge of this research is to identify an interactive adaptive mechanism using 3D modeling and simulation for computer-aided education. Some results of this project are presented below.

STATE OF THE ART
The problem of simulation-based surgery education efficiency is actively discussed in scientific community. Intensive application of simulation tools for surgery training at medical universities and specialized training centers requires implementation of the most up-to-date technologies in robotics, 3D modeling, electronics and software engineering. Still the combination of the most realistic visualization of surgery field and adequate haptic feedback of manipulators that simulate surgery instruments can appear to be not enough for effective educational process. At the same time some simple versions of training suites that are compara-
tively cheap and easy in use can give a significant improvement to a surgery training technique.

For example, in terms of educational outcome complex virtual simulators can appear to be less effective than simple models. The papers (Zhou 2012, Rodgers 2009, Munz 2004) target a problem of looking for a balance between the best simulation and effective utilization of available technologies in practice. The paper (Bello 2011) summarizes recent developments, technologies and actual educational needs and postulates that the educational value of simulation and training technologies changes as a surgeon progresses through his (or her) training, and introduces the concept of virtually augmented environment for different types of training through continual educational process. Thus training should be organized adaptively and individually through interactive tasks. Scientific progress requires difference of educational technologies (Owens 2013). Up-to-date surgical training should use effectively all of simulation capabilities, game technologies, haptics, and virtual environments. (Karaliotis, 2011) highlights the main advantages of virtual technologies in surgery: evolved and realistic human anatomy with normal and pathological conditions and a structured learning environment with controlled levels of difficulty.

Technical aspect can be characterized by an extensive use of emerging and innovative technologies to provide high realism of visual scene and force feedback. Among the most widespread areas of simulation for surgery training there are laparoscopy, laparotomy and endovascular diagnostics and surgery (McClusky 2008, Karaliotis 2011). The major solutions in this area are based on visualization of 3D scenes that represent surgery fields for different cases and simulation of surgery intervention by means of specifically designed manipulators. The concept is pretty close to gaming simulation: the student has a certain situation described by visual model with predefined features and can perform a number of actions getting the response that simulates the real human body behavior.

Due to high complexity and uncertainty specific for real surgery intervention and uniqueness of every individual surgery scene the simulation is often simplified: the number of cases is limited to typical ones (normal and pathologic), and the possible behavior of the model is captured by a certain scenario. This helps implementing the simulation of not all the organs of a human body with realistic physiology and feedback, but only a fragment relevant to the current surgery case. Therefore the student can perform a limited number of actions at a certain moment of time; otherwise the system will terminate the game with an exception.

There should be mentioned that each type of simulators has its own niche. Training suites without force feedback are constructively simple and relatively cheap. They allow surgeons to gain basic surgery skills and can help to evaluate a theoretical qualification and career potential of a medical student who is going to perform certain surgery interventions. Those that provide force feedback are able to offer a new range of educational techniques and bring the student closer to real life practice. One of the main problems here is to train the students to operate surgical instruments, which requires studying uncommon hand motions.

State of the art analysis and consultancy of medical professors and doctors form a number of leading medical universities results in a conclusion that in the area of surgery education there remains a variety of methodologies and educational technologies. From IT service point of view this gives reasons for a relevance of SDK development that can be used as a platform for new simulation solutions. This is a challenging technical problem: SDK should provide interoperability, compatibility and usability. To provide these features there was developed an architecture, were the data (3D models, scenes and scenarios) are separated from UI and core components.

SDK SOLUTION ARCHITECTURE

To provide high flexibility of the developed technologies and enforce its practical use there was developed SDK architecture (see Fig. 1), which contains a number of components that can be used to implement a large variety of simulation solutions for surgery training. SDK contains the components to adjust the educational scenarios, a universal module platform, and cloud services for technical support and professional communications.

This architecture forms the base of new simulation training suites. Most of the already released training suites and the ones that are planned to be delivered in the nearest future use the same approach of physical simulation, 3D visualization and force feedback manipulation. Therefore, these common modules should be separated and implemented with a high degree of universality of applications. As every up-to-date information system the training suites support distributed architecture with cloud services and local information environment build on top of data exchange system.

To implement this architecture there was developed a set of human body 3D models and a number of algorithms of their implementation in training suites. Developers of a new laparoscopy training solution can build their software using the existing manipulators with documented API. After being delivered and integrated into the data exchange system this new solution gets access to a special service capable of downloading new 3D models published by external developers and using them in training scenarios.
**BASIC 3D MODELS OF HUMAN BODY**

3D models of human body parts form the information basis of surgery simulation training SDK. These models set up the input for visualization core components that can generate laparoscopic or endovascular scenes close to real images processed in the process of surgery intervention. Therefore there was no opportunity to use any existing models and a new specially designed set of models was developed from scratch.

By the moment of this paper presentation there were designed, developed and conjoined up to 3000 models of human body parts (see Fig. 2 – 3) combined to 12 layers of human body, including the ligaments, blood vascular system, innervations system, outflow tracts, lobar and segment structures of internals. To develop the models there were used real digital volume computer tomography and magnetic resonance tomography images.

There was also delivered a shell viewer and a database of 3D models of human body. The resulting solution (see Fig. 4) attracted high interest of a sufficient number of professors at medical universities in Russia and is currently available in the market under the product trademark “Inbody Anatomy”.

**BUILDING 3D SCENES FOR ENDOSCOPIC AND ENDOVASCULAR SIMULATORS**

The described above 3D models of human body parts were used to develop a number of scenes for laparoscopy and endovascular training suites (see Fig. 5 – 6). To provide realistic picture these models were fashioned with the help of specifically designed shaders and some fragments like jars, liquids and blood were implemented in software. These shaders and algorithms were implemented apart from the models; and any new models including those that are im-
plemented by third party developers can be used to generate surgery scenes as well.

To provide highly realistic visual and physical models there were used PhysX (Nvidia PhysX) and Bullet (Bullet physics library) middleware. The inner parts of a human body are simulated in the scene by soft body models and surgery instruments are simulated by rigid bodies. The value of the feedback force is calculated on the basis of current geometrical position of soft and rigid bodies in the scene considering their deformation and/or topology distortion.

Both PhysX and Bullet are popular amongst 3D game developers and allow the development of highly realistic simulation scenes. These engines feature 3D collision detection, soft body dynamics, and rigid body dynamics. Additional efforts were needed to overcome some challenges that appeared in practical usage of these engines for 3D surgery interventions simulation.

- a special approach was introduced to implement occlusion queries in simulation of images received from a video camera – endoscope;
- new scenes were developed to train basic surgical skills by using two endoscope instruments and endoscope simultaneously;
- special models were developed to simulate liquids like blood, bile and water and their interaction with soft and rigid bodies;
- some new approaches were suggested to ensure simulation scene dynamics and functioning in real time;
- several scenarios and different cases were developed, by which the student can train in different surgical techniques, learn to react to incoming events and correct possible mistakes.

The software was developed in C++ using .NET C# as a platform for user interface and infrastructure support.

The developed software provides realistic simulation of surgical intervention with the usage of different laparoscopic instruments and video camera – endoscope in 3D scenes. There were 3D models of the human body and operative intervention scenarios also developed that help the learning of basic skills and techniques of surgical treatment.

**SOFTWARE AND HARDWARE COMPONENTS FOR SURGERY SIMULATORS**

Hardware partials like laparoscopy manipulator, camera-endoscope etc. were also insulated and reorganized as autonomous components with specific API providing interoperability with other training suites modules.

To provide realistic physical feedback there was designed a new construction of training suites and developed special software to simulate the process of operative surgery. To simulate the movement of different surgical instruments an original construction was developed. The main assembly unit is the same for all types of manipulators as they differ only in mount attachment, orientation to console and types of adjustable portable handles.

The laparoscopic manipulator was built using 4 Dynamixel actuators (Robotis Dynamixel site) that provide four degree-of-freedom feedback and movement. The main feature of the construction is that the entire outfit is integrated into the manipulator. This helps to reduce any transmission and as a result increases robustness and feedback sensitivity as compared with widely spread analogs. Such a solution entailed a necessity to develop specific algorithms of feedback force calculation based on transition of the physical model to forces and positions simulated by manipulators.

Optical sensors are used to capture the movement of camera-endoscope, and electronic actuators are introduced to simulate the movement of manipulators and provide force feedback. Such construction allows the introduction of different positions of manipulators and simulates cholecystectomy, hernioplasty and gynecology. Special scenes and
study methods are provided to train basic surgical skills. The developed laparoscopic training suite use is presented at Fig. 7.

INTELLIGENT TECHNOLOGIES FOR SURGERY SIMULATION

The main problem of simulation surgery training can be stated as the most effective utilization of interactive technologies to provide fast and stable learning. This can be formalized by two objectives: minimum average learning time with limited deviation and good average evaluation results (high grades estimation) just after the simulation and after certain check time interval.

It should be noted that simulation technologies can be used at different stages of surgery education: for students’ introduction and basic skills training, and for advanced training and skills improvement of experienced doctors. Besides, the simulation training suites can be used together with new surgery equipment for familiarization. Due to the difference in use cases and target user groups as well as the distinction in professional skills and theoretical background of students the training scenarios and estimation procedures should be individualized. As soon as it is impossible to determine the students’ level before the training the simulation scenarios should be adapted in real time.

Top-level solution vision is illustrated by Fig. 1. Surgery training suite software provides a set of training exercises on the basis of surgery case and scenarios description, 3D models, and estimation (evaluation) technique.

Users perform the exercises using the components of training suite software and hardware. In addition to this regular functionality we propose to introduce 2 additional modules for identification of students’ expertise on the basis of their faults retrieved from the training log and adaptation of exercises in real time. Such an approach provides an interactive procedure of interactive technologies application in simulation surgery training. Identification of students’ expertise allows considering the human factor and adapting the complexity of scenarios for different skill level of students, different types of learning and cognitive abilities, speed of training etc. Therefore training suite should conform to the student’s skills level. It could force him to raise his skills interactively.

To represent and describe the surgical cases in laparoscopy and endovascular simulators there was proposed to use Lua script language. This has made it possible to separate code from the knowledge about the user’s behavior, specify possible deviations from the standard surgery procedure and give the user an opportunity to fix the mistakes.

An ontology knowledge base was designed to capture basic medical knowledge about the surgical intervention stages and possible actions. In case a new event occurs that corresponds to the action in the operating scene the system checks possible actions and proceeds to the next stage. In order to provide the required performance level, the sensitivity of the operating scene is limited and differs at different stages depending on the specific skills being trained at that moment.

The idea of game mechanics incorporation is based on the following principle. The complexity of performing of a certain training case is implemented as a dynamic value. Game logic module analyzes a consequence of student’s performed actions, considering the history student’s previous actions on the case and real time events. In terms of this data, Game dynamics module increases the complexity of certain train-
ing case for this student. For instance, if the student surpasses certain time threshold and his mistakes count on the case passing goes to zero at the same time, the complexity is increased by changing the calibration coefficients.

Furthermore, the level of feasibility is increased by means of incorporation of random component to the training case logic, thus making student prepared to unexpected complications of surgery intervention. According to this approach the consequence of student’s actions in two different episodes of a certain training case can be significantly different. As a result the studying cases become not predetermined: the student knows nothing about the current state except the clinical picture description and needs to perform a surgery intervention.

The student’s mark given by a surgery simulator is usually a grade calculated on the basis of a number of key performance indicators (KPI) characterizing intrinsic factors like human body damage and blood loss and efficiency metric, which is evaluated on the basis of agility, length of the instruments’ movement track and intervention duration. In order to introduce the features of game mechanics the overall grade can be calculated in the form of a score summarizing the evaluation of student’s efforts and giving him an opportunity to improve his skills.

It is proposed to calculate the total deviation in time between the moments of expected actions and the moments of factual actions executed by the student. This KPI can be utilized either as a component of the summarizing grade, or as a separate indicator used by the training system to adapt the complexity of the scenario to the current student. It generalizes the idea of game mechanics being applied to the process of training based on simulation. It is proposed to adapt the complexity of simulation to the student’s intermediate success.

For example, in case the student is novice to work with manipulators and surgery scene he will spend much time on each action, even in case he is good in theory and knows the correct procedure. The system will identify it and simplify the case (by e.g. extending the accuracy of manipulators). In reverse for a student with enough experience of operating with manipulators there can be introduced additional random events (like e.g. unexpected effusion of blood). The ways of simplification and types of unexpected events can be easily specified for certain surgical cases.

Such an approach allows obtaining the following benefits:
- the training suite becomes user friendly and adapts to different students with various experience and entry level;
- the simulated cases attract attention and interest of students in real time;
- the complexity of cases increases accordingly the student’s progress in training;
- the monotonicity of training decreases by means of incorporation of random component to the case script;
- during the process of simulation the fragments of scenarios that are familiar and therefore boring are reduced;
- the system combines the periods of getting pleasure from study and satisfaction from the achievements, which is critical for a good training suite.

**SIMULATION OPERATING THEATER**

The proposed approach allowed developing a number of training suites; together with the products of other providers it allowed to equip a unique simulation center capable to support surgery training at various disciplines and educational programs. The simulation center (see Fig. 9) was established at Samara State Medical University and is currently launched to practical use.

![Fig. 9: Simulation Center at Samara State Medical University](image)

The proposed approach allowed to capture and reuse a number of developed simulation technologies to produce new solutions for medical education. Fig. 10 – 12 illustrate some of such solutions developed on the basis of 3D surgery simulation software development kit.

Fig. 10 – 11 represent the extended version of 3D Atlas “Inbody Anatomy” that can be deployed on interactive board and used for extensive study of human body parts and under the framework of open surgery training suite. Using the interactive technologies in addition to realistic simulation provides simple but effective educational facilities that can be used at different stages of anatomy and medical education: from higher schools to universities.

Fig 12 presents the solution based on utilization of Virtual Reality (VR) headset using head mounted display for surgeon decision making support in the process of surgery intervention. The system called “Autoplan” is being developed for automated control and scheduling of surgery intervention.
Intelligent scheduling is performed on the basis of personalized 3D models of human body derived from individual digital volume computer tomography and magnetic resonance tomography images. The system provides the unique functionality of decision making support that can be used for surgery simulation and control in real time that helps reducing the time of intervention and risks of postoperative complications. The solution can be used for surgery education as well.

High perspectives of 3D Surgery simulation software development kit application in medical simulation education prove its practical utility and motivate further developments in this area.

CONCLUSION

In this paper there is introduced a Software Development Kit (SDK) to provide IT developers a platform to build new simulation technologies for medicine. The proposed approach allows the developers of new training suites to better specify the requirements, concretize the scope, prepare effective tests and improve training efficiency. The basic challenge of this research is to identify an interactive adaptive mechanism using 3D modeling and simulation for computer-aided education.

Identification of users’ expertise and the corresponding adaptation of training scenarios make the simulated surgery intervention fascinating, i.e. attract the student’s interest in the process of acting. Surgery exercises can be performed without trainer’s assistance and the average learning time is minimized preserving the same learning results. The simulating suite is used both for study and examination.
ACKNOWLEDGEMENT

This research was financially supported by the Ministry of Education and Science of Russian Federation (grant 2014-14-579-0003), contract 14.607.21.0007.

REFERENCES


AUTHOR BIOGRAPHIES

Anton Ivaschenko, Doct. of Tech. Sciences, Professor at Samara State Aerospace University

Alexandr Kolsanov, Doct. of Medical Sciences, Professor, Director of the Institute for Innovations, Leader of the Research center “IT in medicine” at Samara State Medical University

Aikush Nazaryan, PhD student at Samara State Medical University

Andrey Kuzmin, PhD, Assistant Professor at Penza State University
AFFECTIVE COMPUTING: MEASURING THE PLAYER EMOTIONS IN VIRTUAL REALITY ENVIRONMENTS

Gabriel B. S. M. Moreira, Paulo R. C. Faustino, Matheus M. Ramalho, Alex F. V. Machado, Matheus F. O. Baffa and Esteban W. G. Clua
Department of Computer Science
Federal Institute of Education, Science and Technology of Southeast of Minas Gerais
Av. Dr. José Sebastião da Paixão, Lindo Vale, Rio Pomba – MG
Brazil
E-Mail: gabrielzimmartins@gmail.com

KEYWORDS: Gaming; Man-machines Interfaces; Psychology; Virtual Reality; Simulation Interfaces.

ABSTRACT

Knowing the pleasure a game can provide to the user, might be the key to boost your sales. However, accessories such as a head-mounted display allows the experience with immersion, but it still has a very poor evaluation. Because of the increasing demand of games in this category, this paper presents many applications developed with Unity3D engine, integrated with virtual reality equipment such as Oculus Rift, in order to conduct a study about the player’s emotion. It was used the Emotiv Epoc+, a low-cost mobile EEG recorder, in order to capture, on real time, the interest level, stress and emotions felt by the player and a qualitative evaluation through interview. While testing in different scenarios, this paper shows the strengths and the weakness of the games developed for this type of equipment.

INTRODUCTION

Through virtual reality, presented in different types of electronic games, you can have many experiences related to the similarities of the virtual and the real world. According to [Cardoso et al., 2007], virtual reality is defined as a computational system used to create an artificial environment that allows the user to observe and interact with many elements. The authors also add that when a user is immersed in a virtual environment, he can get different feelings that traditional interfaces cannot provide.

Immersion can be conceptualized as the situation in which a system brings the perspective of the real world to the user. (de Oliveira 2013) says that this concept initially reached many fields of study, such as literature and cinema, and after the twentieth century, it could be observed in virtual reality. According to (Kiner and Siscoutto 2007), virtual reality can have different concepts, assuming that the premise corresponds to immersion and electronic games, there is a definition that describe it as a system that connect the real and the virtual world giving the idea that they exist in the same space.

Many available devices provide the feeling of immersion in a game. In this paper, we are going to use both Oculus Rift and Emotiv Epoc+ to study and observe the emotions provoked in the player.

AUGMENTED VIRTUAL REALITY

According to (Boas 2013), virtual reality (VR) is an abstraction of computing that aims to create a virtual world in order to make the experience as real as possible. Through virtual reality, present in different types of video games, you can acquire different experiences on approaching virtual reality to real reality. According to (Cardoso et al. 2007), virtual reality can be defined as a computer system used to create an artificial environment that allows the user to observe and interact with the elements of the environment. The authors add that when a user is immersed in a virtual environment, he can experience different sensations than traditional interfaces cannot provide.

Another way to approximate the user and computer software is using augmented reality, which can be defined as a method of joining the virtual and the real world. The difference between Augmented Reality and Virtual Reality is that while VR needs equipment for its visualization, such as a monitor, a projector or a headset, and is usually used in closed places, different from AR, which does not require equipment restrictions and can be used in both closed and open environments being more universal, as (Kirner and Siscoutto 2007) said.

The focus of virtual reality is to bring entertainment to the users, but the creation of this new field aimed to reach other areas. The violent conflicts of the twentieth century were events that brought great discoveries to the world and also started the development of virtual reality. With virtual reality, it was possible to create different ways to train soldiers, simulation projects, such as war vehicles and weapons simulation. There were also projects in other fields of studies outside military, but with less investment (Boas 2013).
IMMERSION

Immersion, according to (Boas 2013), is the sensation of being deeply involved in certain activity. In virtual reality for example, you will be able to feel inside the game. There are three types of immersion systems: non-immersive, semi-immersive and totally immersive. Non-immersive systems are the basic systems to simulate the virtual reality, such as desktops which are cheap and do not require too much performance; the totally immersive systems, which are systems, which are the systems more closed to the reality using high quality graphic and require a high performance; and the semi-immersive systems or hybrid systems, which is the combination of the non-immersive with the totally immersive system. An example of it, would be the flight simulations, which combine the high performance software with stereoscopic vision, increasing the immersion experience.

Virtualization

As the definition 5at (Mahalil et. Al 2014), virtualization is a process where the user will be able to have a 3D view with the highest level of immersion. In the case of therapies, virtualization would be a better way to treat people with a disorder. Because of the VR, the therapist will have a better control of what the patient will see. It also helps the therapist view the problem at the root of it, with images of what the patient is seeing and the sensations he is feeling in the treatment.

DEVICES USED

The most traditional input device in computer is keyboard and mouse. With them, the user can access all the functions in the computer, but this paradigm is being left behind in the games. According to (Boas 2013), in virtual reality, data input devices typically track the user's movements, making it more intuitive. This topic will present some of these technologies.

Head-Mounted Display (HMD)

HMDs devices are those with lenses that allow the user to have a broad view in 3D and have the camera movement according to the head position of the user. As said in (Boas 2013), accelerometers, gyroscopes and others sensors can do this movement track.

Oculus Rift

To achieve a higher level of immersion, we used the Oculus Rift as viewing option of our applications. Being an HMD, Oculus Rift has lenses that project a 3D environment and allows tracking of the user's head movements, allowing the mouse movement is made from this. With this technology, we could lead experiments to measure the user’s emotions while playing the game.

Brain Computer Interface (BCI)

BCI is a direct communication between the brain and a device associated with it. These devices function transmitting signals from the brain to the application, for processing and analysis of the data.

Emotiv Epoc+

Epoc+ is a BCI device, which has 14 sensors placed in the user’s head to capture electromagnetic waves produced by the electrical activity of the brain cells. It also has wireless headset and its frequency can reach 128hz says (Duvinage et al. 2012).

A very important feature of Emotiv Epoc+ is to be able to detect the user’s emotion. These emotions are: Meditation, Frustration, Excitement and Engagement. This tool helps to understand better the stress levels, relaxation, and excitement among other sensations that the player is feeling.

(1) Meditation: when the user is concentrated in some task.
(2) Frustration: the reaction of when something unexpected happens.
(3) Excitement: the level of interest of stimulus of the user.
(4) Engagement: related to the attention on what is happening.

Mesuring Emotions

There are many different ways to measure the human emotions. On (Salovey and Mayer 1990) work, is it said that the emotional intelligence are related to monitoring the feelings and the emotions, differentiating them and using this information to guide other people's action. The aspects that involve this perspective are: evaluation and emotion expression, emotion regulation and the usage of the emotional adaptation. According to the author, it is not easy to measure emotion, since there is no trustable equipment for this measurement. The instruments used to be based on self-report collecting the opinion of the person who was being tested, said (Bueno et al. 2003).

In (Harrison 2013), he proposed a study of the accuracy of Emotiv Epoc+ comparing the emotions detected by Epoc+
and the user’s self-report. In his project, he showed pictures to seven different people and compared the emotions reported by Epoc+ and in the self-report. The Epoc+ device did not showed itself 100% accurate to determine the some emotions of the person.

For this project, we worked with both self-reports and Emotiv Epoc+ data to make an accurate measurement.

**RELATED WORK**

Augmented reality is present in many different applications, from entertainment to health, and it is growing up fast nowadays. In this section we are going to approach some related works similar to our paper.

*Snow World*

Snow World is a virtual reality application which use Oculus Rift to show an environment covered with snow developed by (Hoffman et al. 2014). The goal is to use as pain relief for patients being treated for burns, reducing the amount of drugs used in their treatment. During the burn treatment, the patient interacts with snowmen and igloos, using a wireless mouse to target and shoot snow balls (Hoffman et al. 2014). The snow scenario isolates him from the real world relieving the pain.

**DEVELOPED APPLICATION**

*Freefall*

In this scenario, the player is on a platform above a fictional city where he is free to move and is able to choose jump when he wanted. After decide the best time to leave the platform the player begins to fall heading toward the city entering in the immersive scene with wind and sound support, bringing more reality to the game and causing fear, dizziness and other sensations in the player.

*Walls Closing*

The player is at a dark room where in front of him is just a long hall with walls that over time will be tapering, the character in the case may not move, the movement is made by the application and the character moves by itself reaching out to the end of hall. In this application the despair and anxiety to get out of that place is seen as some feelings that we want to provoke in the user.

*Room Underwater*

In this game, we created an environment where the player is in a drowning situation. The character is in an enclosed area with a strong water leak, flooding the area until the character gets completely covered by water. The immersion in this game is made by the sound of the falling water and the water view up through the body, provoking fear and anxiety.

*Platform above the fire*

The main goal of this game is to test the user’s psychological, simulating a platform above the lava with fire particles and smoke. Sometimes the sensation of heat is a psychological factor, which affects the person in many ways, such as when he is nervous or anxious and start feeling warm and sweating. Likewise, we expected that this application provoke the same reaction in the user.

*Spider*

In this game, our goal was to surprise the player. We simulate an environment with a spider positioned behind the player, or when he looks around the map, he gets amazed and surprised by the presence of a giant spider.
EXPERIMENTS

We used Emotiv Epoc+ to detect the player’s emotions while he was playing the game. After that, we showed a list with one hundred possible feelings and emotions and asked to select three options for each application.

The graph on Figure 9 is related to the person who had a previous experience with games and virtual reality. It is possible to see that his excitement level rises only during the fall, since this is the only moment he was not familiar with. The frustration is minor compared to the other participant, since this one presents a previous knowledge on computer games.

CONCLUSIONS AND FUTURE WORK

The goal in this project was to explore the player’s emotions, having previous contact with games or not. With these experiments, we can conclude that it is very important to be aware of the feelings that the player feel while he is playing a game, first to increase his level of satisfaction and then to ensure the product quality avoiding bad experience or to cause wrong sensation while the player plays the game. For future works, it can be done some modifications in the games, to provoke and study different emotions in the player and change the scenarios to apply this study in a specific field of study, such as neural science or psychology.

ACKNOWLEDGEMENTS

The authors acknowledge the Programa de Educação Tutorial of the Ministério da Educação (MEC/SESu/Brazil) for the financial support guaranteed for the accomplishment of this work.
REFERENCES


BIOGRAPHY

GABRIEL B. S. M. MOREIRA studied computer science at Federal Institute. He is a research assistant at Programa de
A SIMULATION BASED GAME APPROACH FOR TEST DRIVE EXAM

Rula Al-Azawi
Gulf College, Muscat, Oman
E-mail:rula@gulfcollegeoman.com
E-mail: rulaalazzawi@yahoo.com

Aladdin Ayesh
DMU University, Leicester, UK
E-mail:aayesh@dmu.ac.uk

KEYWORDS

Agile Methodology, Game Development Methodology, Simulation Game, Multi Agent Software Engineering (MaSE), Agent Oriented Agile Based Game Development Methodology (AOAB).

ABSTRACT

Simulation game usually has a positive impact on training results. This advantage of simulation game lays to get a safe training environment, where the users are able to play, test and enhance their results. It is important to engage learners by providing a motivation and challenging environment related to the real work tasks. With the use of a participatory simulation game approach, we have developed a simulation game to drive test game in order to help the players to get a virtual training environment before they apply for driving test license.

We have based on AOAB game development methodology to implement our simulation game. As the first step, the concept paper and Game Design Document(GDD) is defined, then the analysis and design phases are addressed. In those phases, the QSEE software is used to create the required AgentUML (AUML) diagrams. In the implementation phase, 3DUnity game engine is used for game implementation.

Our simulation game is divided into three main parts: traffic sign exam, Drum or parking exam and road drive exam. The results go beyond the game creation and point to consideration regarding simulation games presented in educational contexts and are suitable for industry sectors.

1 INTRODUCTION

Serious and simulation games are useful for teaching people how to interact with each other and with their environment. The best serious games are simulations that have the appearance of a game, but whose events or processes are real. Usually they include business domains or military operations; many popular entertainment games are based on business and military operations, but with simpler rules [1].

In this paper, our case study that implements simulation game covers the whole life cycle of AOAB methodology, from requirement specification to the game release.

Simulation games are more enjoyable and fun when they provide sufficient challenge for the player [2]. Therefore, important aspect of the game is the game development methodology that will be used in the game design and creation. For this reason, we have selected AOAB methodology as our game development methodology.

AOAB methodology is a hybrid methodology consisting of predictive model using Agent Oriented Software Engineering (AOSE) methodology and adaptive model by using Agile methodology [3]. Furthermore, AOAB combines agile methodology that meets the dynamic requirements of the customer with AOSE which is a rapidly development area of research designed to support development of complex and distributed system in open and dynamic environments with the use of intelligent component. Game development methodology works better when we used iterative methodology because it allowed to have the features ready and to discover and work the fun of the games easier [4].

The reminder of this paper is structured as follows. Section 2 presents a brief overview of computer simulation games. Section 3 describes the game development methodology which has been used in our game creation. Section 4 presents the proposed simulation game with full description. Critical analysis has been presented in section 5. Finally, section 6 presents the conclusion and future work.
2 COMPUTER SIMULATION GAME

Computer games are classified according to genre. Some common genres include action, strategy, adventure, role-playing, sports, simulations, and classic puzzle or board games [5]. Deyo et. al. [6] affirms that all games involve some form of simulation. This is very true, due to the fact that all game developers make considerable effort to recreate, to a high degree of verisimilitude, some aspect of the real-world. However, in contrast to other game genres, simulation games do not always involve a specific goal oriented activity within the context of the game. Games such as Sid Meier’s Civilization [7] belong exclusively to the simulation game genre, as pursing a goal-oriented activity is optional in the game. FreeDictionary [2005] defines simulation games as "mixtures of games of skill, chance, and strategy that result in the simulation of a complex structure". Simulation games can be participatory, iterative, procedural, or situational in nature.

In participatory simulation games, physical simulation places the player within the setting of the game itself. To make progress in the game, the player is then required to perform actions in the game under fictional circumstances. In digital games, examples include virtual reality and augmented reality game systems. Dance Dance Revolution [8] is a good example of a computer game that employs participatory simulation.

In procedural simulation games, an action-consequence model is used to reenact a real-life process. Procedural simulation games are designed to train users to complete a process by following a well-documented set of procedures. Most training simulators fall into this category of simulation games.

The next section will explain AOAB game development methodology which has been used to design our simulation game.

3 GAME DEVELOPMENT METHODOLOGY (AOAB)

The relation between serious games and the AOAB proposed in this research is that the agent can play the roles of adversary and collaborator in a serious game. The serious games can entertain, but its primary goal is to educate, investigate or advertise. In particular, this research is oriented to a category of simulation games called game-learning, the main objective of which is training [1].

The main aim of creating “Drive Test game” is to demonstrate how the actors, scenes, context and game environment could simulate, specified, design and development using the AOAB. The agent needs to be informed about player characteristics and use them in the game play control, and the simulation must feature engaging comments that will motivate the player to play and learn [9].

AOAB combines agile methodology that meets the dynamic requirements of the customer with MaSE which is a rapidly development area of research designed to support development of complex and distributed system in open and dynamic environments with the use of intelligent component.

Game development methodology worked better when we used iterative methodology because it allowed us to have the features ready soon and to discover and work the fun of the games easier. Ideally, the type of hybrid development methodology approach which we already defined in AOAB is recommended for use by independent game developers. This possesses a mix of characteristics that would sit somewhere between those of a predictive or adaptive approach to be generic methodology useful for small or large game projects.

Agile methodology is usually used to deal with dynamic changes in requirement specification by the customer, customer involvement in the development phases. For the flexibility in adding new requirements even before game release which does not add extreme cost to the project, Agile game development methodology will be adapted to suggest game development methodology as adaptive model [10].

AOSE provides such intelligence through agents. Agent may perform the tasks individually. In complex and distributed system, Agents can be used to monitor the interaction among components and to interact as human interaction. The MaSE used in the Sprint phase is at the core of the AOAB [11]. Each iteration includes analysis, design, implementation, testing and evaluation of MaSE as shown in Figure 1.

The reason to adapt MaSE is that it is the core of Agile as in complex systems and distributed systems such as games, it is difficult to trace a single point of control, since the objects are distributed [12].

4 PROPOSED SIMULATION GAME (DRIVE TEST GAME)

The driving test game is an educational game with a graphic format. The player controls the character and this character interacts with different environments and situations and must make decisions. The game is made up of several scenarios, or locations, based on the Oman learn driving center website [13], where the driver must perform certain actions correctly to pass the three parts of the driving exam.
4.1 REQUIREMENT SPECIFICATION

The first step of the game creation is to write the requirements scripts. This script is usually called a concept paper [14]. The game designer will convert the information from the concept paper to a GDD, which will serve as a guide throughout the development process. The last steps of the requirement specification are to create a Sprint Backlog based on the GDD, which should provide an estimation of the number of iterations needed prior to game release.

4.1.1 GAME CONCEPT PAPER

The driving test game is a simulation game. The goal of the game is to simulate the environment of the driving test that must be passed to obtain a driving license for automatic or manual cars. The environment of the game will be split over three areas, based on information from [13], which contains basic information required to know and understand how to drive safely, particularly for the road test, drum test or parking test and, finally, the traffic sign test.

The current web page [13] includes most of the scenarios that a driver could face. The proposed game will be more useful and understandable than any PDF file or web page. Furthermore, the language of the web page is in English only, which, in the Sultanate of Oman, many people do not understand well. In the game proposed by this research, both Arabic and English languages were used.

The first environment of the game is the traffic sign test. Usually, the test presents 5 to 8 signs. The game will ask for 5 signs at random, with an option to view all signs and their meanings. The second environment is the parking test or drum test, which deals with how to park a car between two rows of drums, or between two cars. When the driver parks in a handicapped parking space, or touches a drum or another car, the game will highlight the incorrect parking. The third environment is the road environment, where the driver is sitting with an examiner and following their orders on a real road. This is the hardest part of the driving test, as the examiners sometimes give the driver incorrect orders to follow. The real test takes approximately 15 minutes and the same time limit will be applied in the game. The driver will sometimes need to refuse the examiner if wrong orders are given, for example if the examiner asks the driver to park in handicapped parking; if the driver accepts this order then they will fail the test.

4.1.2 GAME DESIGN DOCUMENT (GDD)

For most designers, GDD is a fun and interesting activity, as they are able to apply the vision that was presented in the concept paper. The complete GDD is not an easy piece of work [15]. A poorly elaborated GDD can lead to a need for reworking and loss of investment in game development phases. Therefore, this research will analyze several available GDDs found in existing literature, comparing the findings to propose an improved general GDD placed alongside a commercial GDD. Most authors agree that there is no established structure for a GDD, as there are significant differences among games. However, there is a set of common elements of a game design [16]. These common elements are used to create a general GDD as a template that is easy to use and can be applied to different game genres. Et. Jesse Schell [17] suggests creating more than one document to serve all necessary purposes. Schell defines six groups, which need to remember and communicate different things, as shown in Figure 2. The GDD template is constituted of one document divided into multiple subsections.

![Figure 2: Game Design Document Elements [17].](image-url)
4.1.3 SPRINT BACKLOG

The final step in the requirement specification of the AOAB is to translate the GDD into a Sprint Backlog. The workload was distributed among the team during the Sprint planning meeting. In this meeting, it was decided to divide the work into four iterations, where each iteration would cover one environment of the game as shown in Table 1.

4.2 GAME ANALYSIS AND DESIGN

The analysis phase is a very fruitful period; it depends mainly on what is collected in the requirement specification phase. The first step in the analysis phase is capturing goals, which usually depends on the requirement specifications phase and transforms into a structured set of system goals, depicted using a goal hierarchy diagram for the drive test game. In the goal hierarchy diagram, the main goal is defined, which is less likely to change than detailed steps and activities. The goals are organized by importance. Figure 3 illustrate sample of our goal hierarchy digram. We have used Agenttool3 software which is linked with Eclipse software to create all required diagrams for the analysis and design phase.

![Diagram](image)

Figure 3: Main Goal Hierarchy

4.3 GAME IMPLEMENTATION

This section will discuss the actual implementation, using 3Dunity as a game engine. Our implementation section has been divided into subsections according to the game layout, game animation and game programming as will be explained in the next section.

<table>
<thead>
<tr>
<th>Table 1: Time Plan of Our Game</th>
</tr>
</thead>
<tbody>
<tr>
<td>Requirement phase</td>
</tr>
<tr>
<td>Analysis phase(Sprint1)</td>
</tr>
<tr>
<td>Design phase(Sprint1)</td>
</tr>
<tr>
<td>Implementation phase(Sprint1)</td>
</tr>
<tr>
<td>Analysis phase(Sprint2)</td>
</tr>
<tr>
<td>Design phase(Sprint2)</td>
</tr>
<tr>
<td>Implementation phase(Sprint2)</td>
</tr>
<tr>
<td>Analysis phase(Sprint3)</td>
</tr>
<tr>
<td>Design phase(Sprint3)</td>
</tr>
<tr>
<td>Implementation phase(Sprint3)</td>
</tr>
<tr>
<td>Testing and evaluation phase(Sprint4)</td>
</tr>
</tbody>
</table>
4.3.1 GAME LAYOUT AND GRAPHIC

The initial step in the game creation is to import the car model to the 3DUnify working area, as shown in Figure 4. The car model is an important part of the game design. In this work, some traffic sign images are imported to form part of the traffic sign test. Furthermore, in Figure 5 some sample traffic signs are included along with their Arabic and English descriptions. Our game considers different levels based on our game design which was divided into four Sprint phases. Therefore our main screen in the game also includes many level selection as shown in Figure 6. Actually, multiple messaging scenes are used to communicate with the player and explain how to pass the driving test. Figures 7, 8 and 9 illustrate the results of certain actions, if they are correct or if they are wrong, and also provides the results of the test.

4.3.2 GAME ANIMATION

To achieve a perceptible representation of the dynamical movement of a car, the car control Java script is added to the car model to provide movement for the car. Two box Colliders are used for the car, as well as a wheel Collider to the four car wheels. A simple way to check collisions using unity is by adding a Rigid-
body component. Rigidbodies are physically simulated objects that can be used as marks. A way of using Colliders is to mark them as a trigger. For this research, it is useful for triggering a specific event in game. Furthermore, another camera is added to follow the car’s movements. As a first step, another camera is created and imported using a package named Script to use the Smooth follow Java file, and link it with our car. To achieve a better viewing experience for the player, the distance between the camera and the car is minimized.

4.3.3 GAME PROGRAMMING

As described earlier, 3DUnity has the ability to deal with Java Script, C sharp and Boo programming languages within the same project. Java Script, as shown in Figure 10, is used to move from one scene to another within the 3DUnity project. Each time the game moves between scenes, the scene name appears in the last line of Figure 10. Figure 11 is a Java script imported from the scripts package. A Smooth follow Java file allows a second camera to follow the car and enable a clear view for the player. The Java script in Figure 12 shows a case in which the player selects the correct option on traffic sign test; his/her final score will increase by one.

In the case of an incorrect selection, the score will not increase, as seen in Figure 13. Figure 14 illustrates the final score for the player after answering all questions in the traffic sign test. The Java script is imported from Scripts with name car control to add control to the car, as shown in Figure 15. Each of the previous Java Scripts must be linked with a scene or a component of a scene in order to work perfectly. Usually, if the component details are selected, the name of the linked Java file can be found. The next section will discuss the actual steps for implementing the game, based on the previous analysis and design section.

5 CRITICAL ANALYSIS

Our simulation game provides a complete test and evaluation of AOAB methodology which covers the academic and industry requirements for game creation. This is an interesting experiment from the perspective of teamwork and knowledge sharing. The team is small and the communication excellent. Our team were happy to create a detailed GDD that saved a lot of time at the implementation stage. Furthermore, the analysis and design diagrams gave team members a clear view of what needed to be done, how it should be done, and by whom. Finally, it is clear that AOAB enhanced the progress of the game in relation
to the final game release. AOAB provides powerful documentation, which is useful for the game evaluation and the creation of new versions of the game.

It was observed that the game engine considerably increased the productivity of the game design, and 3DUnity was adequate for the project’s development needs. It is easy to create executable files for desktop computer platforms; furthermore, 3DUnity does not require much effort to work with multiple platforms.

One of the research aims was to use the AOAB methodology in the industry sector and to increase the quality of the game prior to final release, but there are still some limitations highlighted by our team. It was required, after each iteration, to integrate the work, but this can sometimes create a problem if certain steps are not performed in the correct way. In some cases, there were problems regarding diagrams in new game iteration.

6 CONCLUSION AND FUTURE WORK

The drive test simulation game is mainly designed based on AOAB methodology. Our team involved were glad to have access to our AOAB as it allowed consideration of the management section. The schedule, budget and user satisfaction are the main goals of any commercial game company, and AOAB deals with all of these important points in a clear and systematic way. Most games companies avoid dealing with complex methodology; they typically use clear steps and easy to follow phases, which appear clearly in the AOAB methodology. Furthermore, one of the main and critical prob-
lems in the industry sector is feature creep; AOAB provides a solution to this problem by including the customer in each game iteration, as well as complete analysis and design for games, covering all game requirements and needs. The players were glad to get an experience of driving game and to check their experience in traffic sign and how to park the car before the actual exam.

REFERENCES


VIRTUAL PRODUCTION ENVIRONMENTS
Semi-real evaluation environments for mobile applications

Sascha Alpers, Christoph Becker, Esmahan Eryilmaz
FZI Forschungszentrum Informatik
D-76131 Karlsruhe
Germany
E-Mail: {alpers|becker|eryilmaz}@fzi.de

KEYWORDS
evaluation environment, mobile applications, evaluation infrastructure, context

ABSTRACT
In this paper, an innovative, semi-real evaluation environment is presented that supports the evaluation of interactive, context-sensitive mobile applications. This hybrid approach uses plasticlal and digital simulation to build a suitable evaluation environment that allows for more efficient evaluation than comparable approaches. Business Process models are utilized to synchronize state information and control commands between the plasticlal and the digital model. Additionally, the approach supports the integration of generated synthetic information in order to enrich the evaluation environment and mimic a multitude of different evaluation cases.

1. INTRODUCTION
Ubiquitous mobile applications are used within highly dynamic and very heterogeneous environments. The application’s behavior depends not only on the used hardware, but also on the properties of applications, systems, users, and infrastructure currently surrounding it. That means, while dealing with problems, crashes or issues on mobile applications, there remains the key problem of detecting the environmental conditions in which the problem or task occurred. If those are not completely transparent to the evaluator,² wrong conclusions and interpretations may result. Therefore, it is important to gather information about the application’s behavior in various (extreme) environments for which it has not been designed and developed. Subsumed, that means one should evaluate the application’s behavior and features in expected environments as well as within special complex situations. Complex situations, for example, might comprise low bandwidth in combination with low battery and highly loaded servers. It is essential to evaluate the application within such complex combinations of different environmental aspects.

² The goal of the evaluator is not to prove that the solution works as this has to be done during development by the developers and testers of the solution. The evaluation is done after deployment and should figure out how well the solution meets the original intended goals or how the solution behaves in use.

A convenient concept for this kind of complex evaluation and validation environments is offered by so-called Living Labs. "A Living Lab represents a user-centric research methodology for sensing, prototyping, validating and refining complex solutions in multiple and evolving real life contexts" (Eriksson, Niitamo, und Kulkki 2005). At FZI House of Living Labs (Hellfeld 2014), founded in Karlsruhe in 2012, a research environment with 2,000 square meters over two floors has been built up — today comprising of 8 Living Lab — dealing with a range of different research challenges on information and communication technologies. The FZI Living Lab mobileIT/mobileBusiness is working on new methods, technologies and solutions to integrate mobile information technology into business processes of enterprises in an efficient and sustainable way.

For the special purpose of evaluating interactive and context-sensitive mobile applications, a new approach based on a semi-real evaluation environment has been developed and integrated into the FZI Living Lab mobileIT/mobileBusiness. Within this environment, the following components are relevant:

- A plasticlal model, as representation of the real environment, in which the application will be used later on.
- A digital model, as representation of the data in real environments.
- A mediating platform, which is linked up to the application and used to simulate various real world states.
- A configuration and manipulation application, installed on the same device as the application we are going to evaluate.

In the following sections, the approach as well as the features of the dedicated components, will be described in detail. The remainder of the paper is structured as follows. Section 2 describes modeling in general terms as models and building models is an essential step for building up evaluation and simulation environments. Modeling is an abstraction process and in order to build the environment, one has to abstract from real world scenarios that are relevant for the mobile applications. In section 3, simulation basics and simulation environments will be described as basic work for the approach presented in this paper. The details of the evaluation environment, as well as the term semi-real environments and it’s relation to existing terminologies like augmented, mixed and virtual reality, are introduced and discussed in section 4. Afterwards, section 5 describes the
approach including existing and planned components. Finally, the paper closes with a conclusion in section 6.

2. MODELING
In general, models are a simplified reflection of a model origin for a definite purpose (Kaschek 1999). A model origin could be a contrived system (which does not exist yet) or a part of the real world (e.g. an already existing artefact). For different purposes it can be necessary to reflect different aspects of the model origin. This can be done by building up different models for certain views or perspectives of the same model origin.

Take, for instance, an airplane. Perhaps there exists a digital aerodynamic model to simulate aerodynamic aspects through computers; as well as a 3D-printed aerodynamic model to test these aspects in a wind tunnel; to train the pilot (especially for exceptional situations), there is also a flight simulator needed; and for the training of the cabin crew, a special cabin model is also used. All these artefacts are models, each one representing different aspects of an airplane – the same model origin. They are built for different purposes with different methods, and they are all different views of the same origin.

Depending on the purpose, models can be represented as text, graphic, sculpture, or a mathematical structure. In many cases, modeling (performing the reflection) is done using modeling languages. Modeling languages define the syntax of the model and the semantic of different element types. A modeling language helps structuring a model in order to support a common understanding.

Process modeling, as a part of business modeling, describes activities within a process and rules that determine in which sequences the activities ought to be executed (in the different instances of a process). Additionally, it is possible to describe the "entities that flow through the process" (Tumay 1996), such as information objects or control flow tokens. In many cases, additional aspects like references to required resources are integrated into process models as well.

In graphical process models, activities are represented by nodes that are linked through arcs. Most modeling languages define special nodes for alternative or concurrent execution of activities which are used to describe the rules for the flow. In the presented approach, we use Petri Nets (see Reisig 2013, or van der Aalst and Stahl 2011) as modeling language with a graphical and a mathematical representation. Thanks to the mathematical foundation of Petri Nets, in addition to simulation [see section 3], it is also possible to perform formal verifications and various analyses. This paper, however, does not focus on such verifications and analyses.

3. SIMULATION AND EVALUATION ENVIRONMENTS

Simulation provides effective and efficient means to analyze and observe a system’s behavior in a certain environment. In this context, the system’s behavior is referred to as state changes over time (Gosavi 2015). In order to simulate and analyze a system’s behavior, the system has to be modeled [see section 2] beforehand. These models can be interpreted and executed by simulation software. Based on the variables and associated probability distributions, the simulation results, i.e. the system’s behavior, vary.

Although the simulation approach is not suited to calculate an optimum solution, it is applied in several domains in order to evaluate different alternatives or scenarios. In Business Process Management, for example, simulation presents a sophisticated approach to analyze business process models (Tumay 1996). Therefore, business process models have to capture relevant constraints and decision rules. Process models are used to digitally simulate processes. That means a computer steps through various instances of the process in compressed time, tracking simulation data that allows the calculation of measurements such as throughput, costs (if assigned to activities), and many more. In contrast to project management tools, spreadsheets and flow charts, business process simulation allows capturing the dynamics of interacting entities, taking randomness and uncertainty of the real world into account.

An additional application of simulation can be observed in the context of software testing, especially regarding ubiquitous mobile applications. These kinds of applications have to be tested and evaluated in target environments (Morla and Davies 2004) because the functionality, as well as certain possibilities of interaction with the application, strongly depend on the current contextual environment. This leads to expensive and time-consuming testing that does not scale. However, software-based simulation provides the means to simulate single aspects of the environment such as network related or location-based aspects so that on-sight testing can be reduced or even omitted.

Although software-based simulation is an adequate choice for many applications, there are complex situations that are hard to grasp. This is especially true for complex real life situations that require context-sensitive human-machine communication, as well as further interaction with physical objects. The Living Lab approach, as introduced in section 1, addresses these scenarios. The Living Lab approach integrates stakeholders (experts from different domains as well as end users) to collectively develop, test and evaluate new innovative solutions. The Living Lab provides an infrastructure to apply, test and evaluate technology in real world scenarios and everyday use.

4. SEMI-REAL ENVIRONMENTS

The prefix ‘semi’, which originated from Latin and is freely prefixed to English words, describes something as being “partly, in some degree or particular” (Stevenson 2010, 1618). Within this paper, ‘semi-real’ describes on the one hand the evaluation environment itself and on the other hand data that is used for the evaluation. The evaluation environment has been built up based on a plastic model, also referred to as ‘semi-real physical environment’, which can be described as the substitute for the future environment.

358
of the app. This aspect of the semi-reality allows the interaction with physical objects. Another aspect, referred to as ‘semi-real informational environment’, covers the partially real informational environment, combining real sensor data with generated synthetic information about the environment. The connection between these two aspects of semi-reality (plastical and informational) is established using an underlying business process model.

As none of the already well-known terms — such as mixed, virtual, or augmented reality — correctly match the particular focus of the presented approach, a new terminology is introduced within this paper. Virtual reality would mean the real world is replaced by a simulated one (Milgram und Kishino 1994), (Steuer 1995), which does not reflect the presented approach correctly. The term mixed reality is inappropriate as well, as the approach does not merge real and virtual worlds equally (Milgram und Kishino 1994). Although the basic definition of augmented reality could match — as it includes the augmentation of real environment with any kind of digital information — the broad understanding of augmented reality focuses on visual information generated in real time and presented to users on displays (Navab 2004), which again does not reflect the key aspects of the approach presented in this paper. To differentiate the approach from these terms and concepts, the presented simulation and evaluation environment is described as semi-real, focusing on the real world situation and its configuration through extra-added synthetic information.

In the following three subsections, these two aspects of semi-reality and their connection via business process models will be discussed in detail. In particular, the paper will focus on the creation of various semi-real views on business process models within one single physical environment, and the opportunities that become possible regarding evaluation purposes.

4.1. Semi-real informational environments
As introduced in the beginning, the presented approach offers the possibility to collect information about mobile application behavior on real context that is enriched with generated context. The combination of real and simulated context is described as semi-reality and a semi-real environment respectively. This digitally generated synthetic information is added to the real world to simulate a specific real world situation or state which is needed to fulfill the requirements of a certain so-called evaluation case. Different evaluation cases can comprise combinations of completely

Figure 1: Plastic Model as 'real' part of the semi-real environment
different types of contextual information, such as real and synthetic information or different quality levels of contextual information — for example, different accuracy levels of location information. The presented approach maps various real-world states onto the evaluation cases which we then set up within one evaluation environment. The evaluator will conduct the evaluation cases. That means that by using the mobile application within the environment, the evaluator is going to observe the applications behavior according to the specification and document it using the tools provided with the presented approach.

Considering the heterogeneity in the mobile device market with widely differing hardware capabilities of devices, and, in particular, a differing information quality of single sensor results, it is obvious that not all relevant evaluation cases can be built within one single physical evaluation environment. Therefore, this paper introduces semi-real environments. The information delivered by real devices as well as environmental sensors, is enriched with digitally generated contextual information and in doing so, simulates semi-real environments, each one matching the requirements of a certain evaluation case. This real world configuration is done with a simulation control app that allows manipulation of sensor information and simulation of special conditions by digitally enriching the real physical environment. A suitable example is the configuration of temperature or the limitation of network bandwidth.

4.2. Semi-real physical environments
In order to provide an evaluation and testing environment for mobile applications with a strong focus on human interaction with both the application as well as surrounding physical objects, it is necessary to interact with those or with adequate substitutes. Therefore, a semi-real physical evaluation environment of a traditional production line has been created within the FZI Living Lab using 'Fischertechnik Robotics' (Fischertechnik 2015) machine models. Every machine is used for the plastical visualization of a special component of the line. As illustrated in Figure 1, the environment comprises six conveyor belts, three 3D robots, two punching machines, two indexed lines (each with two different machining stations and two transports over corners), and two pneumatic machining centers. Each Fischertechnik component has its own integrated controller that executes the commands given to it by the Windows 2003 Server (see 'mediating platform' in Figure 2). The Server runs a tomcat that receives the commands from mobile clients and status information from Fischertechnik components. After receiving a command from a mobile device, the server runs the corresponding java statements and controls the different actions on the Fischertechnik components.

This semi-real physical environment does not only allow the interaction with physical objects in a suitable environment, it also supports the evaluation process itself. The proposed plastical visualization makes it easier for the evaluator to apply a systematic approach and handle different evaluation cases properly within one physical evaluation environment.

4.3. Connection of semi-real physical and informational environments via business process model views

![Figure 2: Semi-Real Evaluation Environment](image-url)
One of the main features of our evaluation environment is the multifaceted connection between a digital process model and the plastic visualization of it. This link offers the possibility to connect and synchronize the informational semi-reality with the semi-real physical environment. The technical connection is implemented by a mediating platform.

The mediating platform monitors the current status of the environment and controls the actions of the machines in the plastic model. That means, firing a transition in a digital model results also in observable actions in the plastic model. At the same time, the mediating platform is responsible for the transfer of digitally generated information to the physical environment. Specific status information about the semi-real environment is mapped to the digital process model through a token as part of the dynamic component of the Petri Net.

5. MOBILE APP EVALUATION IN SEMI-REAL EVALUATION ENVIRONMENTS
In this section the opportunities of the introduced semi-real evaluation environments for mobile app evaluation purposes will be discussed.

The interplay of the single components within the evaluation environment is demonstrated based on a mobile app which is used to perform some tasks within a complex business process of an enterprise. For example, this could be an app which supports the mobile working maintenance staff, or an app for the sales representatives. As already described above, this app will be used in various and dynamically changing environments. Therefore, it is necessary to evaluate how the app behaves while performing the business activities under certain environmental conditions. This task is supported by the semi-real evaluation environment implemented in the FZI Living Lab.

As shown in Figure 2, the mediating platform is linked up to the following: to the plastic as well as to the digital model of the business process; to the digital enrichment app on the test devices; to the simulation control application; to the evaluation monitoring; and of course, to the app that is to be evaluated. The evaluator can perform various evaluation cases by configuring the real environment through the generation and addition of digital context information to the current environment with the help of the configuration app.

At first, the evaluator has to decide which evaluation cases and environmental conditions could be relevant for the mobile app. Therefore, it could be necessary to get support from domain experts. These evaluation cases are the main inputs given to the simulation control component which consists of two subcomponents: the first one is deployed on the test device and responsible for the establishment of the semi-real environment. The second component is generating digital information which is sent to the first subcomponent via the mediating platform.

Monitoring the state information retrieved from the different views on the business process model via the mediating component, the simulation control component can systematically execute the defined evaluation cases. To evaluate the app’s behavior within this certain semi-real environments, all evaluation data is composed within an evaluation result format which includes the evaluation case, information about the real conditions, as well as the generated data and of course, the results the app produced in this certain environmental situation. These evaluation results are ready to be sent to an analyzer component which can also be an independent third party system component.

6. RELATED WORK
A lot of related work has been conducted in the area of simulation and simulation environments. This section will relate some selected pieces to the approach presented in this paper.

In his work, K. Tumay provides an overview on business process simulation (Tumay 1996). Further, he shows that business process simulation is a suitable analyzing approach to address the complexity, especially with regard to the dynamics of business processes. However, Tumay’s focus solely resides on the simulation of digital business processes and does not connect the simulation to real infrastructure. In 2004, R. Morla and N. Davies presented a test and simulation environment to evaluate a mobile location-based application (Morla and Davies 2004), thereby addressing the challenge of complex and expensive onsite testing. Their approach differs from the approach presented in this paper as it focuses mostly on the simulation of network communication and abstractions from further interaction with physical objects.

The approach presented in this paper goes beyond the work mentioned above as it provides a holistic evaluation environment for mobile applications providing, means for the simulation of environmental aspects on the one hand, and integrates possibilities for physical interaction on the other hand. This is similar to the Living Lab methodology described by Eriksson et al. (Eriksson, Nittomo, and Kulkki 2005) However, the presented approach extends this with the concept of semi-reality.

7. CONCLUSION
Within this paper, challenges with regard to testing and the evaluation of ubiquitous mobile applications have been discussed, subsequently highlighting the importance of adequate evaluation environments, as well as the context in which such an application is tested. To solve these issues, the concept of semi-real evaluation environments, based on context monitoring and manipulating, has been proposed. Next, the paper demonstrated how this concept has been implemented in the FZI Living Lab mobileIT/mobileBusiness in order to provide an infrastructure which is supporting the evaluation and testing of ubiquitous mobile applications, in particular applications with a strong focus on human interaction with both the application, as well as surrounding physical objects. The paper highlights that the results of semi-real evaluation and context monitoring can be of benefit to designers, developers and maintainers of mobile applications, as well as processes...
— giving them an idea about the application’s reactions in different environments, and in particular, in those environments for which it has not been designed and developed. Compared with other approaches where only single aspects are simulated, the advantage of the presented approach is that it integrates neatly with the Living Labs methodology, providing a holistic test and evaluation environment with strong interrelationships to the real world. In our future work, further ubiquitous mobile applications will be tested and evaluated in the semi-real evaluation environment. Therefore, simulation and control components of our environment will be enhanced in order to provide capabilities for analyzing monitored data, controlling application-specific, as well as device-specific properties, and for triggering simulated failures.

8. REFERENCES


A GENETIC ALGORITHM WITH SHUFFLE FOR JOB SHOP SCHEDULING PROBLEMS

Fredrik Danielsson, Bo Svensson, Dhanush Reddy
University West, Department of Engineering Science
S-461 86 Trollhättan, Sweden
fredrik.danielsson@hv.se, bo.svensson@hv.se

KEYWORDS
Simulation based optimisation, Genetic algorithm, Job shop scheduling, Random shuffle

ABSTRACT
Job shop scheduling problems are computationally complex combinatorial optimization problems. Genetic algorithms have been used in various forms and in combination with other algorithms to solve job shop scheduling problems. A partially flexible job shop with precedence constraints increases this complex behaviour. There are two main parts to optimizing a job shop, the routing and the scheduling. The objective here is to get consistent optimal makespan using a genetic algorithm. This paper firstly, presents a simulation approach for the considered partially flexible job shop scheduling problem. Which take into account the precedence constraints and reduce situations of deadlock. To solve the partially flexible job shop scheduling problem a genetic algorithm was used and improved. It utilise a genetic crossovers for routing and a new random shuffle feature is introduced for the scheduling. The computational results have shown that the algorithm performs well in terms of finding a consistent optimal schedule for the given problem.

INTRODUCTION
A job shop scheduling problem (JSP) is one in which a set of jobs is scheduled on a set of machines to minimise a certain criterion, subjected to the constraint, that every job has a specific processing order which is fixed beforehand (Zhang, et al., 2009). Flexible job shop scheduling problem (FJSP) is an extended branch of the classical JSP. In the fields of production management and combinatorial optimization, scheduling and routing of a FJSP is an important factor (Xia & Wu, 2005). In FJSP, an operation of a particular job can be processed on a machine from a set of given machines which opens up many different routes or combinations in the scheduling. The combinatorial complexity involved in both scheduling and routing as a whole is computationally challenging due to its NP-hard nature.

The aim of this work is to use simulation based optimisation with a genetic algorithm to reduce the makespan. The 8X8 problem (Kacem, et al., 2002) is used as reference model to be able to compare the result. In this work the consistency of the algorithm was of prime importance. The consistency of the genetic algorithm in reducing the makespan was compared with and without the new random shuffle feature for sequencing, introduced in this paper. A genetic algorithm in its basic form is a random search evolutionary algorithm. When it is used to optimise a combinatorial optimization problem like the 8x8 partially flexible JSP the number of possible combinations that can be obtained are numerous. A problem representation based on (Chen, et al., 1999) and (Zhang, et al., 2009) is used in this work to solve the partially flexible JSP. This representation was devised bearing in mind the precedence constraints considered in the paper. A detailed explanation is given in this paper. A random shuffle feature is introduced with the GA for scheduling the problem to increase the variability of the search.

BACKGROUND
Different approaches have been used to solve JSP problems. In (Zhou, et al., 2009) the performance of an ant colony optimization (ACO) approach for a dynamic JSP is investigated. They tested the steady state performances of three ACO intermediate measures namely the makespan, mean flowtime and mean tardiness and compared the results with those from dispatching rules such as first-in-first-out (FIFO), shortest processing time (SPT) and minimum slack time (MST). Experiments were conducted to find the best performing dispatching rule as well as find the best ACO measure. A GRASP (Greedy randomized adaptive search procedure) algorithm to optimise the Flexible JSP was introduced by (Rajkumar, et al., 2011). They considered the problem with limited resource constraints with algorithms and hence minimize the makespan, maximal machine workload and total workload of the machines. Flexible JSP with its two sub-problems which are the routing and scheduling problem has also been inspected with other techniques like in (Moslehi & Mahmam, 2011) using a hybridization of the particle swarm optimization namely the multi-objective particle swarm optimization (MOPSO) and local search. They have proposed an algorithm in which the MOPSO algorithm updates the particles based on the obtained values and the local search algorithm further improves these results. They have also compared the efficiency of these algorithms with similar ones such as the weighted summation approach and the pareto approach. A hybrid of PSO using tabu search is used to optimize the multi-objective FJSP in (Zhang, et al., 2009). The information exchange between particles in the proposed approach happens by a crossover operation. Minimisation of makespan, critical workload and total workload was the main goal of this approach, as seen in a few other optimization approaches as well. Hybrids of genetic algorithms and other evolutionary algorithms have also been
used to solve the flexible job shop scheduling problem. A genetic algorithm controlled by an assigned model through approach by localization was used in (Kacem, et al., 2002). One of the first attempts at the FJSP using a genetic algorithm was made in (Chen, et al., 1999). They introduced a new method of representing the chromosomes for routing as well as sequencing of operations. In this paper we propose a method based on (Chen, et al., 1999) and (Zhang, et al., 2009) for representing the problem, a genetic algorithm for routing and a random shuffle function for scheduling is used.

** (PROBLEM FORMULATION) **

The formulation of the FJSP is similar to that of JSP with a few added assumptions and constraints. In this work a partially flexible JSP has been formulated. The set of all jobs \( J = \{j_1, ..., j_n\} \) where \( n \) is the total number of jobs. Each job \( j_k \) consists of a predeterimined sequence of operations, let \( O_{ik} \) be operation \( g \) of job \( i \) (Xing, et al., 2009). The set of all machines \( M = \{m_1, ..., m_k\} \) where \( k \) is the total number of machines. All machines \( m_p \in M \) and all jobs \( j_k \in J \) are available at the beginning, time \( t = 0 \) (Kacem, et al., 2002). For each operation \( O_{ik} \) there exists a nonempty set of machines \( M_{ik} \subseteq M \) capable of performing it. All jobs \( j_k \in J \) and machines \( m_p \in M \) are independent from each other (Xing, et al., 2009). This means that there exists no precedence constraints between operations of different jobs and all the machines can work simultaneously. Non pre-emptive condition is valid i.e., an operation \( O_{ik} \) cannot be interrupted during its execution. Only one operation can be performed on each machine at a given time. Let \( C_p \) constitute the completion time of machine \( m_p \). The objective function is given by \( f \), where \( f \) denotes the makespan. Since the makespan is the total time taken to process all the jobs of the schedule, the longest \( C_p \) time is the makespan,

\[
f = \max_{1 \leq p \leq k} C_p.
\]

Thus, the optimisation problem can be formulated as

\[
\min f(x) \quad x \in S
\]

where the solution \( x \in S \) and \( S \) is the bounded search space.

** REPASSWORDOMENT OF INDIVIDUALS **

The representation of the problem in this paper is partly based on the work of (Chen, et al., 1999) and (Zhang, et al., 2009) wherein the chromosome is represented using a vector namely the aString. A genetic algorithm is used to solve a problem with this representation to increase the variability and randomness of the algorithm. Since it is a genetic algorithm the aString vector can be considered a chromosome. The aString is a row vector, which assigns the operations to the set of machines available for the process. Consider a simple flexible job shop with two jobs having three operations each, which have to be assigned to a set of three machines, aString = [12 13 11 23 22 21]. In the aString row vector the units place in every element represents the machine on which the operation of the job will be executed. The tens place however consists of a sequence of job numbers which avoids creating a deadlock schedule when an operation is replaced by a corresponding job index (Zhang, et al., 2009). In the aString example of a simple flexible job shop, the sequence of job numbers is read from left to right as 1 – 1 – 1 – 2 – 2 – 2. This can be translated into a list of operations: \( O_{11} – O_{12} – O_{13} – O_{21} – O_{22} – O_{23} \) (Zhang, et al., 2009).

** SIMULATION FUNCTION **

This is a mono-objective problem and, the objective function value, the makespan for each solution are represented by chromosomes (Chen, et al., 1999). The makespan calculation is a simulation function, that returns the processing time for the last Position, on a machine, and the function keeps calling itself until the processing time for all dependant Position on the respective machine is calculated. Position refers to the column number of a particular operation in the aString. Processing time refers to the time units required for that operation to be completed on the machine it is assigned to in the aString. On each recursion of the simulation function the Position is changed. It starts from the last operation of each job, which keeps calling itself with its previous operation's Position or the Position of any other operation, which might be a dependency for that operation. The Position of the operators in the aString is translated into a matrix named Position reference for calculating the makespan. Every element in this matrix corresponds to a Position in the aString. The number of rows in Position reference matrix denotes the total number of machines in the job shop where the row number denotes the machine number. The number of nonzero columns for every machine represents the operations scheduled on that particular machine. An example of a Position reference matrix is

\[
\text{Position reference} = \begin{bmatrix}
3 & 6 \\
1 & 5 \\
2 & 4
\end{bmatrix}
\]

The pseudocode explains the method to obtain processing time for every Position i.e., every operation \( O_{ik} \). The processing time for the last operation of every job \( j_k \) is obtained. The highest value of these processing times gives the makespan for that particular aString.

** PSUEDOCODE:**

```plaintext
# EVAL_FUNC

get Position in Position reference matrix for which time is required;
if (Position is first column of Position reference matrix)
    if (Position is first operation of job j_k)
        return execution time;
    else
        return execution time + execution time of [Position - 1];
else
    if (Position is first operation of job j_k)
        return execution time + execution time of Position in previous column;
    else
        return max (execution time + (execution time of [Position - 1], execution time of Position in previous column))
```

364
PROCEDURE

The process of reproduction in a genetic algorithm (GA) generally consists of four steps namely selection, crossover, mutation, replacement (Chen, et al., 1999). In this paper an additional step called the random shuffle is introduced after the crossover and before the replacement stage. The stepwise procedure then becomes:

i. Selection
ii. Crossover
iii. Random shuffle
iv. Replacement
v. Mutation

Selection

Selection is the first step of GA, which is used to select the parent chromosomes required for reproduction. It is a process used for operation assignment. The job-shop has n jobs where every operation $O_{ij}$ of job $j_i$ can be performed on a set of machines $M_{ig}$ as seen from the problem formulation. Two chromosomes i.e., two aString vectors are chosen for reproduction, one being the minimal processing time chromosome and the other chromosome being random. Minimal processing time chromosome indicates that for every operation $O_{ij}$ the machine from the machine set $M_{ig}$ with least processing time for that operation is assigned. The random chromosome is one in which an operation $O_{ij}$ is assigned to one machine from the available machine set $M_{ig}$ at random. These two chromosomes reproduce to give offsprings.

Crossover

Crossover is a process used to generate a better solution by exchanging information contained in the current good ones (Chen, et al., 1999). There are various types of crossover operators such as the single point crossover, uniform crossover, two-point crossover etc. For our purpose we use two-point and uniform crossover. In a two-point crossover, see Figure 1, two positions are selected at random across the aString. Two new aString vectors are created by swapping all characters between and inclusive of these two positions (Chen, et al., 1999). A uniform crossover, see Figure 2, with a crossover probability $p_c$ equal to 0.5 implies that every position has a 50% probability to be chosen for crossover.

Random Shuffle

The new step proposed and introduced in this paper. The order of the operations on a machine, bearing in mind the no precedence constraints, plays a major role in reducing the makespan. The random shuffle function is used for operation sequencing on each separate machine $m_p$. In the algorithm, two random job numbers are chosen after preserving the machine order and their positions are swapped. An example illustrate the purpose of using random shuffle. Consider the aString = [11 13 12 23 21 22] which has two jobs, each job having three operations, whose operations are assigned on three machines. This can be translated to an operation sequence table that shows how each operation $O_{ij}$ of job $j_i$ is sequenced on machine $m_p$ from the machine set $M_{ig}$ for the given aString:

\[m_1: O_{11}, O_{22}\]
\[m_2: O_{13}, O_{23}\]
\[m_3: O_{12}, O_{21}\]

Preserving the machine order for the operations before shuffling is an important step in this process. To illustrate this, first, the aString is shuffled without preserving the machine order and then by preserving the machine order. Shuffled without preserving the machine order gives aString = [11 12 13 21 22 23]. The operation sequence table for this aString can be written as:

\[m_1: O_{11}, O_{21}\]
\[m_2: O_{12}, O_{22}\]
\[m_3: O_{13}, O_{23}\]

In this operation sequence table the machine assignment for operations of jobs when compared to the sequencing table for the aString before shuffling has changed. The purpose of random shuffling is to change the order in which the operations $O_{ij}$ are sequenced on the machine $m_p$ and not to change the machine completely. In order to achieve this, considering the representation of the problem used in this paper, we preserve the machine order before shuffling the operation $O_{ij}$ in the aString. Consider the aString = [11 13 12 23 21 22]. The sequence of job numbers for this aString read from left to right is 1 - 1 - 1 - 2 - 2 - 2 and the machine order for the operations of job $j_1$ and job $j_2$ are 1 - 3 - 2 and 3 - 1 - 2 respectively. The sequence of job numbers is shuffled and the new sequence reads 1 - 2 - 2 - 1 - 1 - 2. When the machine order from jobs $j_1$ and $j_2$ are used in the same order for the new shuffled job sequence, the aString = [11 23 21 13 12 22] is obtained for which the operation sequence table is:
\[ m_1: o_{11}, o_{22} \]
\[ m_2: o_{13}, o_{23} \]
\[ m_3: o_{21}, o_{12} \]

As seen in the first and third operation sequence tables, operation \( o_{12} \) is assigned to the same machine \( m_2 \) in both cases because the aString was shuffled after preserving the machine order. However, the order of execution of operations on machine \( m_3 \) has changed after shuffling.

**Replacement**

Replacement is a selection mechanism of the survivors. After every generation, a replacement mechanism has to be implemented in order to continue with reproduction for further generations to attain the optimal solution. In this algorithm, to increase the variability and randomness for more effective searching, after shuffling and calculating the fitness value of the offsprings, the generation is replaced with one best parent chromosome which returned the best fitness value in that generation and the second chromosome being a random chromosome selected similar to the one at the selection phase of the GA. The random parent chromosome increases the search space by assigning the operations to different machines every time and hence increasing the probability of getting out of the local optima.

A certain fitness value is defined at the beginning until which the replacement strategy continues. Once this fitness value is reached, instead of choosing one best parent chromosome and random chromosome each, two best parent chromosomes are chosen to continue reproduction for further generations.

**Mutation**

Mutation is a process where one or more values of the chromosome are altered in order to achieve variability in the search space. It is an important aspect of the genetic search, which helps in avoiding hibernation of the search at local optima. After the replacement stage in every generation, one best parent chromosome is survived for reproduction in further generations. At the mutation stage, an operation \( o_{i\ell} \) is chosen at random from this chromosome and its machine assignment \( m_\ell \) is changed to a different value from its allowed machine set \( M_{i\ell} \). Consider the aString = [11 21 22 13 12 23]. In this aString the first operation of job \( j_1 \) is assigned to machine \( m_1 \). During mutation stage, let’s say \( o_{11} \) is mutated and it is assigned to machine \( m_2 \) from the allowed machine set \( M_{11} \). After mutation, the aString changes to [12 21 22 13 12 23]. In the new aString, the machine assignment for \( o_{11} \) has changed from \( m_1 \) to \( m_2 \).

**Computational Results**

The implemented algorithm was tested for a partially flexible job shop scheduling problem taken from (Kacem, et al., 2002) which is an 8 jobs x 8 machines problem with 27 operations, see Table 1. It was implemented on a personal computer with a 2.4 GHz Intel core i7 processor. MATLAB is used for the implementation of the algorithm. The instances marked \( X \) suggests the inability of the operation to be processed on that particular machine.

<table>
<thead>
<tr>
<th></th>
<th>M1</th>
<th>M2</th>
<th>M3</th>
<th>M4</th>
<th>M5</th>
<th>M6</th>
<th>M7</th>
<th>M8</th>
</tr>
</thead>
<tbody>
<tr>
<td>J1</td>
<td>011</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>3</td>
<td>X</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>012</td>
<td>10</td>
<td>X</td>
<td>5</td>
<td>8</td>
<td>3</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>013</td>
<td>X</td>
<td>10</td>
<td>X</td>
<td>5</td>
<td>6</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td>J2</td>
<td>021</td>
<td>5</td>
<td>7</td>
<td>3</td>
<td>9</td>
<td>8</td>
<td>X</td>
<td>9</td>
</tr>
<tr>
<td></td>
<td>022</td>
<td>X</td>
<td>8</td>
<td>5</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>023</td>
<td>X</td>
<td>10</td>
<td>X</td>
<td>5</td>
<td>6</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>024</td>
<td>10</td>
<td>8</td>
<td>9</td>
<td>6</td>
<td>4</td>
<td>7</td>
<td>X</td>
</tr>
<tr>
<td>J3</td>
<td>031</td>
<td>10</td>
<td>X</td>
<td>7</td>
<td>6</td>
<td>5</td>
<td>2</td>
<td>4</td>
</tr>
<tr>
<td></td>
<td>032</td>
<td>X</td>
<td>10</td>
<td>6</td>
<td>4</td>
<td>8</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>033</td>
<td>1</td>
<td>4</td>
<td>5</td>
<td>6</td>
<td>X</td>
<td>10</td>
<td>X</td>
</tr>
<tr>
<td>J4</td>
<td>041</td>
<td>3</td>
<td>1</td>
<td>6</td>
<td>5</td>
<td>9</td>
<td>7</td>
<td>8</td>
</tr>
<tr>
<td></td>
<td>042</td>
<td>12</td>
<td>11</td>
<td>7</td>
<td>8</td>
<td>10</td>
<td>5</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>043</td>
<td>4</td>
<td>6</td>
<td>2</td>
<td>10</td>
<td>3</td>
<td>9</td>
<td>5</td>
</tr>
<tr>
<td>J5</td>
<td>051</td>
<td>3</td>
<td>6</td>
<td>7</td>
<td>8</td>
<td>9</td>
<td>X</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>052</td>
<td>10</td>
<td>X</td>
<td>7</td>
<td>4</td>
<td>9</td>
<td>8</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>053</td>
<td>X</td>
<td>9</td>
<td>8</td>
<td>7</td>
<td>4</td>
<td>2</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>054</td>
<td>11</td>
<td>9</td>
<td>X</td>
<td>6</td>
<td>7</td>
<td>5</td>
<td>3</td>
</tr>
<tr>
<td>J6</td>
<td>061</td>
<td>6</td>
<td>7</td>
<td>1</td>
<td>4</td>
<td>6</td>
<td>9</td>
<td>X</td>
</tr>
<tr>
<td></td>
<td>062</td>
<td>11</td>
<td>X</td>
<td>9</td>
<td>9</td>
<td>9</td>
<td>7</td>
<td>6</td>
</tr>
<tr>
<td></td>
<td>063</td>
<td>10</td>
<td>5</td>
<td>9</td>
<td>10</td>
<td>11</td>
<td>X</td>
<td>10</td>
</tr>
<tr>
<td>J7</td>
<td>071</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>6</td>
<td>7</td>
<td>X</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>072</td>
<td>X</td>
<td>9</td>
<td>X</td>
<td>9</td>
<td>11</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>073</td>
<td>X</td>
<td>8</td>
<td>9</td>
<td>3</td>
<td>8</td>
<td>6</td>
<td>X</td>
</tr>
<tr>
<td>J8</td>
<td>081</td>
<td>2</td>
<td>8</td>
<td>5</td>
<td>9</td>
<td>X</td>
<td>4</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>082</td>
<td>7</td>
<td>4</td>
<td>7</td>
<td>8</td>
<td>X</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td></td>
<td>083</td>
<td>9</td>
<td>9</td>
<td>X</td>
<td>8</td>
<td>5</td>
<td>6</td>
<td>7</td>
</tr>
<tr>
<td></td>
<td>084</td>
<td>9</td>
<td>X</td>
<td>3</td>
<td>7</td>
<td>1</td>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>

The parameters like number of chromosomes, number of generations and number of random shuffles were tuned over a certain range for a period of time. For example, the number of chromosomes was tuned in a range between 50 and 500, the number of generations was always high, usually 1000 to 2000. Since the aString has 27 operations the number of random shuffles for one chromosome was varied between 2 and 50 random shuffles. The algorithm was executed a total of 70 times with a population of 200 chromosomes, 1000 generations and 13 random shuffles with the crossover, shuffle, mutate and replacement operations. The average makespan value obtained after these runs was 18.16 with a standard deviation of 1.01, see Figure 3. The x-axis indicates makespan and y-axis indicates the number of times that makespan was obtained during the 70 runs. The algorithm was tested for different values of number of chromosomes, generations and number of random shuffles. The mutation rate was kept constant throughout the tests. Mutating more operations in the aString did not make a difference in the consistency of the fitness value. The algorithm was also tested without the random shuffle feature resulting in an average makespan value of 19.30 with a standard deviation of 1.56, see Figure 4. These results
CONCLUSION AND FUTURE WORK

The aim of the paper was to reduce makespan of a given partially flexible JSP using simulation based optimisation and a genetic algorithm. Genetic crossovers control the routing of the problem and the scheduling is controlled by a new proposed random shuffle function. The algorithm was tested with and without the new random shuffle function. The genetic algorithm with random shuffle has shown to achieve better fitness values over the one without the random shuffle consistently. This can be seen in the computational results, see Figure 3 and Figure 4. The genetic algorithm with the new random shuffle produced an average makespan of 18.15 over the one without random shuffle, which produced an average makespan of 19.30. Also the standard deviation was better with random shuffle, 1.01 compared to 1.56 without random shuffle. This implies that scheduling or sequencing of operations on the machine plays an important role in reducing the makespan. Future work will be done on this algorithm by devising a more controlled shuffling technique for scheduling. Other algorithms could be incorporated to make the shuffling intelligent, to further improve the consistency and speed of convergence.

REFERENCES


ROBOTICS SIMULATION
SIMULATING AND IMPLEMENTING THE INTEGRATION OF SERVICE ROBOTS INTO MANUAL WORK PROCESSES

Frank Heinze  
Dept. Robot Technology  
Frank.Heinze@rt.rif-ev.de

Maisie Kloeckner  
Dept. Production Automation  
Maisie.Kloeckner@rif-ev.de

Juergen Rossmann  
Dept. Robot Technology  
Rossmann@rif-ev.de

Bernd Kuhlenkoetter  
Dept. Production Automation  
Bernd.Kuhlenkoetter@rif-ev.de

Jochen Deuse  
Dept. Industrial Engineering  
Jochen.Deuse@rif-ev.de

Institute for Research and Transfer, RIF e.V., Joseph-von-Fraunhofer-Str. 20, 44227 Dortmund, Germany

KEYWORDS
Manual processes, MANUSERV, industrial service robotics, Industry 4.0, planning, robot program generation.

ABSTRACT
The introduction of service robots in industrial environments is the starting point for the new robotics area of industrial service robotics. Characteristic feature of industrial service robots is the cooperation or even collaboration of service robots with humans in an industrial environment. The number of real industrial service robot applications remains relatively low, as it is difficult to analyze the economic impact of transferring a manual process to a collaborative process. The research project MANUSERV directly addresses this challenge. The objective is to develop a planning and decision support system for transferring manual processes to automatic or semi-automatic solutions. We compare the proposed solution with the original manual process in a simulation environment. Another aim is to generate source code for the chosen service robotic system. Three industrial research partners provide use cases in the form of manual processes for the evaluation of the developed system. In this paper, we focus on the generation of source code and the subsequent modeling and simulation of the different use cases in a combined way in a demonstrator setup. The demonstrator setup inside the simulation environment includes manufacturing stations of different automation degrees, humans and service robotic components. Moreover, we introduce a method, which shows that it is possible to use the results of the simulation tool platform-independent.

INTRODUCTION
Currently, we see a radical change in manufacturing technology; a fourth industrial revolution seems to take place right now. The first industrial revolution took place with the invention of automatic looms driven by waterpower or steam engines. The usage of electrical energy for automation and the introduction of flow production by the usage of assembly lines characterize the second industrial revolution. The third industrial revolution began with the usage of electronic systems and IT technology for automation purposes and the use of robots and PLCs for assembly processes (Dombrowski and Wagner 2014). The current fourth industrial revolution is on its way with the internet-of-things, cyber-physical systems (Schlick 2012) and the collaboration of humans and robots as industrial service robots. In the German political and scientific discussion, this potential disruptive change is termed Industry 4.0 (Hirsch-Kreinsen and Weyer 2014). Still, manual processes are quite common in many areas. Actually, it is extremely difficult to decide if a manual process has the potential for transferring it to an automated or semi-automated process, where an industrial service robot supports the human. The objective of the research project MANUSERV (from MANUal processes to industrial SERVice robots) (Deuse et al. 2014) is to develop a planning and decision support system to evaluate the automation potential of manual processes. Besides the decision support relating to an optimized automation system, the implementation of the chosen robotic components in the regarded processes is another overall objective. The approach consists of several consecutive steps, starting with a detailed description of the manual process. The user inputs this description into a planning system that proposes automatic or semi-automatic processes instead, if this is possible. The planning results serve as input for a simulation of the newly designed process. The simulation helps to decide if the process is technically feasible. In order to simulate the process a source code generator translates the output of the planning system to the robot language of the recommended service robot system. This source code can then drive the real service robot, too.

The presented approach describes how to combine human motions and service robotic components into a manual production process with the help of simulation. Moreover, it presents an approach how it is possible to create generic coding for different robotic systems.

The paper starts with a short introduction of the state of the art of task planning for service robots and continues with a description of the general system concept with the focus on providing a platform-independent source code and generating robot-specific source code. We will describe the reference processes and their simulation and explain the simulation of the derived demonstrator setup including elements of all reference processes. Finally, we will present the results and give a further outlook of the approach.
TASK PLANNING FOR SERVICE ROBOTS

The idea of automatic task planning for service robots was also addressed in the research project DESIRE (Plöger et al. 2008), which stands for German service robotic initiative. The research project used the quasi standard PDDL (Fox and Long 2003) as planning language. One of the results of DESIRE was that a simulation system would be a useful enhancement. In MANUSERV, the simulation system is one of the central building blocks. Even if the results were rather promising, the DESIRE system never reached a state that enabled its professional use. Robot programming by demonstration (Siciliano and Khatib 2008) is very close to this approach. Possible solutions here watch a human operator via a camera system and try to reproduce the human action or reproduce the human movement by executing movements generated by sensors attached to the human body.

The idea of interconnecting programming and planning was previously addressed (Lang and Zanuttini 2012) and later on improved such that classic planners can generate computer programs with control flow statements (Jiménez and Jonsson 2015).

SYSTEM CONCEPT

Manual processes as they are currently found in many areas are the starting point for MANUSERV. Concerning these manual processes, the objective is to decide if it makes sense to replace the manual process by an automatic or semi-automatic process. Figure 1 shows the general system design of MANUSERV. The idea is to start with a detailed description of the manual process with a standardized description language. The user inputs the description via a web page interface into a planning system to generate the initial state, the goal state, and a list of all possible actions. The planning system gets data of the capabilities of different service robotic systems via a database as second input. Thus, service robot manufacturers can input the data of their robots, so that the planning system considers them as possible solutions.

From these inputs the planning system generates as output a recommendation concerning the appropriate hardware configuration and a list of the required actions of the service robots and human operators to get from the initial state to the goal state. The planning system finally splits this action list up into a so-called elementary action sequence. Elementary actions are single actions for which it does not make sense to split them up into smaller units.

The planning and decision support system needs to expand this outcome further. The aim is to include robotic specific parameters into the actions. The parameter values strongly depend on the manual processes. In this manner, it is possible to describe all necessary actions of the robot, which are the basis to generate a platform-independent source code. Figure 2 shows the general steps of the procedure.

![Figure 2: Source Code Generation](image)

Actions describe the process execution. They are defined according to VDI 2860 (VDI 2860), for example “moving”, “grasping” and “depositing”. System-specific parameters and environmental-specific parameters expand these actions. Examples of system-specific parameters are gripper state, velocity, number of axes, accuracy, ultimate load. Examples of environmental-specific parameters are current position, next position, movement type, position of the manipulative object, and additional environmental information.

For the action “moving”, there are different movement strategies like “move linear”, “move joint” and “move circular”. Figure 3 shows the different movement types most often found in robotic systems.

![Figure 3: Typical Movement Types](image)

Every kind of movement strategy has special expressions. Figure 4 shows for example potentially auxiliary conditions for linear movements.

![Figure 4: Auxiliary Conditions for Linear Movements](image)
The user interface of the source code generator helps to adapt the different commands to the considered processes. Before selecting all the necessary instructions and parameters it is necessary to choose for which robotic system the generator should produce source code. After selecting the used robotic system, the entry mask will open for entering system-specific parameters. The entry mask depends on the chosen robotic system.

Over the years, several hundred different proprietary robot programming languages were developed (Hesse 1998). The difficulty is that there is no strict systematic structure in the languages, e.g. lack of grammar. Therefore, these languages are difficult to handle in the same manner (Freund et al. 2001). The only existing standard, the industrial robot language (IRL), which is normed by DIN 66312 (DIN 66312), is not used by any industrial robot. Only the simulation environment CIROS (Rossmann et al. 2010) uses it as one possible robot programming language.

In a first step, we focused on three selected robot languages for the development of a tool, with which it is possible to solve the described problem. Table 1 gives an impression of the source code differences of robot programming languages. Source code examples for a linear movement are shown for KRL (KUKA robots), Rapid (ABB robots) and UR (Universal robots).

Table 1: Linear Movement Commands for Different Robot Types

<table>
<thead>
<tr>
<th>Robot language</th>
<th>Movement to P30</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>UR</td>
<td>Movel (pose, a=1.2, v=0.25, t=0, r=0)</td>
<td>In tool space</td>
</tr>
<tr>
<td>Rapid (ABB)</td>
<td>MoveL robtarget, v500, z50, tool;</td>
<td>Tool is defined</td>
</tr>
<tr>
<td>KRL (KUKA)</td>
<td>LIN P20</td>
<td></td>
</tr>
</tbody>
</table>

The linear movement command example shows that there are differences in the programming languages regarding syntax and semantics, but that a translation of planning actions into robot language commands is generally possible.

In the next step, the simulation uses the generated robot programs to verify the recommended system configuration of the planning system. The simulation checks the functionality of the process and the components. After the simulation, it is necessary to transfer the results to the real robots in a user-friendly way. An important point is the generation of a platform-independent source code for the operator. The presented concept solves this requirement; the elementary action sequence is platform-independent and it is transferable to a specific robot language for simulation.

Our planning and decision support system evaluates different robotic systems as possible solutions for the regarded processes. Even if there are a large number of potential robotic systems, it needs to be possible to generate system-specific source code for each robotic system. A possible result of the planning and decision support is that one of the regarded processes needs more than one robotic system for an optimum system design. This shows the importance of generating a platform-independent source code. This way, it becomes very easy and user-friendly to integrate such components into individual production processes. This results in a higher acceptance rate and less barriers.

REFERENCE PROCESSES

In the project MANUSERV, three industrial application partners provide use cases ranging from manual assembly via machine setup and maintenance to farming application. This broad range of application areas is helpful to keep the developed system as flexible as possible for future application areas. All these use cases represent manual processes where there is currently no straightforward automation solution.

In farming, there are many manual processes still today. Together with our application partner from the farming industry, we decided to select the cleaning of dairy cow cubicles as an application area. This belongs in general to the dairy farming area. In dairy farming, there are already several solutions for the automation of the milking process, but the automation of the cleaning process is generally still relatively low.

For the alleys between the dairy cow cubicles there are two predominant solutions, solid floors and slatted floors. One way to clean the alleys automatically is to use sliders. Another automation solution is the use of a mobile robot called manure robot. Figure 5 shows on the left a typical slider on a solid floor and on the right a manure robot for the cleaning of slatted floors.

![Figure 5: Dairy Farm Automation with a Slider (Left) and a Manure Robot (Right)](GEA Farm Technologies GmbH, Germany)

Sliders or manure robots clean slatted floors by pushing the manure over the slatted floor, so that it falls between slats. For solid floors, there exists only the less flexible solution in the form of sliders that push the manure to either side of the floor.

The cleaning of dairy cow cubicles is usually still a manual task, which takes place at least once or twice a day. There are two basic types of dairy cow cubicles, high bed cubicles and deep bed cubicles. Figure 6 shows high bed cubicles on the left and deep bed cubicles on the right.

![Figure 6: High Bed (Left) and Deep Bed (Right) Cubicles](373)
High bed cubicles usually consist of a rubber mat on a concrete base often with additional litter. Deep bed cubicles consist of concrete beds filled with a larger amount of litter. Typical litter types for high bed cubicles are hay or sand. As we consider the cleaning of high bed cubicles more promising, we focus our automation approach on this type of dairy farm cubicle.

Our second partner develops stretch blow molders for PET bottles. Figure 7 shows a typical stretch blow molder with a bottle transfer system on the left, the heating module in the middle and the blow molding module on the right.

![Figure 7: Stretch Blow Molder for PET Bottles](KHS Coroplast GmbH, Germany)

Manual processes exist mainly in the maintenance and setup area. The most important setup process is a product change. There are two important manual processes during a product change. When a different bottle type is produced, the forms for the bottles have to be changed in the blow molding module. Even more laborious is the product change when the stretch blow molder has to produce a bottle with a different thread diameter. Mandrels hold the bottle preforms on their way through the heating unit. A change of the bottle thread diameter induces the change of every mandrel. As a stretch blow molder can have up to 300 mandrels, this is a very tedious task. Our industrial partner optimized the manual process, but in general, an automation of this process promises a strong reduction of setup time of up to several hours. Therefore, our focus is to find a solution for the mandrel changing of stretch blow molders.

Our third industrial partner has a high number of manual assembly processes. Our use case is the assembly of an electric appliance for in-wall mounting. The assembly is currently done at two manual assembly stations. At one station a worker assembles the power supply, which is generic for all units, at the other station another worker assembles the exterior design, which is highly customer specific. We choose this process because we see a high potential for designing a cooperative process with a manual part and a robot-automated part.

**SIMULATING THE MANUAL PROCESSES**

The simulation of automated and manual production processes and the comparison between manual and (semi-) automated solutions requires a simulation environment providing extensive 3D simulation functionalities. In this case, the simulation environment VEROSIM (“Virtual Environments and Robotic Simulation System”) (Rossmann and Schluse 2011) was used. VEROSIM is capable to simulate human motions by using a model of the human body, the so-called “Virtual Human”. The “Virtual Human” models the human locomotor system as multiple industrial robot arms, simplified to their kinematic chains (Rossmann and Sciclete 2010, Schlette 2012). The simulation of industrial production processes is another main application area. In general, VEROSIM is capable of generating virtual environments for the analysis and optimization of real processes.

In order to compare the manual processes with the (semi-)automated processes generated by the planning and decision support system of MANUSERV it is important to be able to simulate the manual processes as well as the (semi-)automated processes. Our concept for integrating human motion via a motion capture device into the simulation environment (Heinze et al. 2015) forms the basis for the simulation of the manual processes.

Figure 8 shows the simulation of the manual assembly process. The movements of the virtual human worker are the same as the movements of the real operator because we recorded the movements with a motion capture suit.

![Figure 8: Simulation of a Manual Assembly Process](image)

Figure 9 illustrates the simulation of the cleaning process of dairy cow cubicles. Again, the simulation uses recorded movements of a real human operator. Figure 10 gives an example of the simulation of the mandrel changing process.

![Figure 9: Simulation of a Dairy Farm Cleaning Process](image)
MODELLING THE DEMONSTRATOR SETUP

One major goal of the demonstrator setup is the transfer of the results, which are the outcome of the developed planning and decision support system. Prior to the implementation a simulation tool validates the outcome of the planning and decision support tool in detail. This also includes a detailed analysis of the whole layout of the planned automation system. Furthermore, it is the aim to integrate the aspects of the use case applications of the industrial partners including a service robotic solution into the demonstrator setup. Thus, this implementation shows the use of a mobile service robot in various applications.

In the demonstrator setup a single dairy cow cubicle, which the operator or a robot needs to clean and intersperse, represents the dairy farming application. The machine setup includes parts of a stretch blow molder for PET preforms. There the task is to change the mandrels that hold the preforms. Figure 11 shows details of the demonstrator setup with the farming application on the left side and the mandrel change application on the right side. Furthermore, the demonstrator setup contains a manual assembly process with two assembly stations: one for manual assembly and another one for hybrid assembly. Hybrid assembly signifies a human robot cooperative solution. Moreover, the layout of the assembly station enables an examination of the functionality of intermediate and final products. Additionally, the demonstrator setup integrates a stock where the mobile service robot can pick up the components needed at the various workstations.

A possible mobile manipulator, which we can integrate into the demonstrator setup, is MMO - 700 from Neobotix. It combines the omnidirectional MPO - 700 with a lightweight robot arm from Universal robots (UR10). With the functionality of the MMO - 700 it is possible to realize the assembly processes and the machine setup. We will need to modify it to realize the clean and intersperse part in the farming application under laboratory conditions. The properties of the mobile manipulator fit the requirements of the demonstrator setup very well. The integration of the mobile manipulator into the demonstrations setup has the following advantages: The mobile manipulator is able to move in all directions with low vibration and a low noise level, because of the used omniwheels in conjunction with appropriately subdued drives. Thus, a soft and clean movement is possible during the ride. It is possible to let the manipulator assemble some of the parts during its movement between the fixed assembly stations, if it reaches the required accuracy. Another advantage is the open communication interface, which is based on ROS (“robot operation system”). In addition to integrating service robots into industrial processes it is the aim to further expand the human-machine collaboration. Therefore, the simulation needs to implement the interaction between robot and human. It is possible to precondition the inclusion of safety and surveillance functions (Thomas et al. 2011). Even for hybrid work systems, it is necessary to provide support for safety and surveillance functions in advance.

The demonstrator setup shows, that the integration of industrial service robots into manual processes is possible. Furthermore, the user can estimate the additional benefit of the use of industrial service robots by comparing the simulation results of the demonstrator setup with the simulation results of the manual processes. Here, an economic evaluation regarding process time and process cost gives the user the necessary information to decide whether a change from the manual process to the recommended automatic or semi-automatic process is beneficial.

RESULTS AND DISCUSSION

The objective of the research and development project MANUSERV is to develop a planning and decision support system to help the user decide if it is useful to transfer a manual process to an automatic or semi-automatic process and what type of robot or service robot the user should utilize. The approach starts with a detailed description of the process that the user then feeds into the planning and decision support system. Process-specific information has to enhance the output of the planning system, e.g. the positions of work pieces and machines. This enhanced action sequence is platform-independent and ensures an easy portability to different types of robot systems. We use the presented approach to transfer the platform-independent code to specific source code for the different hardware platforms that might solve the automation problem under examination. The
simulation can use the generated source code as well as the platform-independent code to simulate the application, but the generated source code can also drive the real application. An evaluation of the recommended solution concerning technical and economic aspects is possible by simulating all the generated solutions. The comparison of the recommended process with the manual process is possible as we can also transfer the manual process into the simulation environment. The next steps consist of evaluating the presented concepts in the demonstrator setup and finding reasonable criteria for the comparison of the manual and the automatic or semi-automatic processes.

ACKNOWLEDGEMENTS

The research and development project MANUSERV is funded under Grant No. 01MA13011A within the scope of the German “Autonomics for Industry 4.0” technology program run by the Federal Ministry for Economic Affairs and Energy and is managed by the project management agency “Technical Innovation in Business” at the German Aerospace Center in Cologne. We would like to thank our industrial partners ICARUS Consulting GmbH (Lüneburg, Germany), GEA Farm Technologies GmbH (Bönen, Germany), Albrecht JUNG GmbH & Co KG (Lünen, Germany), and KHS Corpoplast GmbH (Hamburg, Germany) for their valuable input and contributions.

REFERENCES


VDI 2860. 1990. Assembly and handling: handling functions, handling units; terminology, definitions and symbols.

AUTHOR BIOGRAPHY

FRANK HEINZE studied electrical engineering in Paderborn, Germany and Urbana-Champaign, Illinois, USA and graduated with a diploma degree in 1996 when he joined the Institute of Robotics Research at the TU Dortmund University, Germany. He has been working for the Institute of Research and Transfer, RIF e.V. in Dortmund since 2005. In 2011 he earned a second diploma degree in business administration and engineering from the University of Hagen, Germany. His research areas include industrial automation, human robot cooperation, virtual reality and graphical simulation systems. Currently, he is project coordinator of the German research project MANUSERV.
Simulation-based Time and Jerk Optimisation for Robotic Press Tending

Emile Glorieux, Bo Svensson, Fredrik Danielsson
Department of Engineering Science
University West
S-461 86 Trollhättan, Sweden
email: emile.glorieux@hv.se

Bengt Lennartsson
Department of Signals and Systems
Chalmers University of Technology
S-412 96 Gothenburg, Sweden

KEYWORDS
Production, Optimization, Manufacturing, Automatic control, Industrial control

ABSTRACT

Increased production rate and robot motion smoothness in a sheet metal press line are essential. Smooth robot motions avoid unplanned production interruptions and excessive wear of the robots. Reaching high production rate and smooth motions requires tuning of the tending press robot control to minimise the cycle time and jerk. Doing this for a press line with multiple robots is a complex large-scale problem. To model such problems for the optimisation process, computer simulations become almost essential. This work presents simulation-based optimisation of the time and jerk of robotic press tending operations and investigates the importance of including the robot motion’s smoothness. An optimiser works in concert with a simulation model of a sheet metal press line and its parametrised control system. The effect of including jerk minimisation in the objective function is tested on a real-world problem concerning a sheet metal press line. The results illustrate the importance of including jerk-minimisation as an objective in the optimisation. Furthermore, the performance of this approach is compared with manual tuning by experienced operators. The results show that the proposed simulation-based optimisation approach outperforms manual tuning.

INTRODUCTION

For high-speed robot operations in many current applications, optimal robot motions in terms of time, velocity, jerk and path is of increasing importance in high-volume manufacturing. This is the case for press tending robots that load and unload products from the presses in automated sheet metal press lines (Svensson et al. 2013, Glorieux et al. 2015). Optimising the paths and motions of press tending robots can significantly improve the line’s production rate (Osakada et al. 2011, Glorieux et al. 2015, Jimenez et al. 2013, Li and Ceglarck 2002). As with many real-world optimisation problems, the press line problem is complex and it also is a large-scale problem because multiple robots in the line must be considered (Svensson et al. 2013, Glorieux et al. 2015).

For complex real-world optimisation problems in manufacturing, it can become necessary to use a simulation model to evaluate the trial solutions (Andradttir 1998). In those cases, it typically is very challenging, cumbersome and impractical to describe the problem as an abstract mathematical formulation to evaluate the trial solutions. Though, building a reliable simulation model is also a time-consuming task that requires specialised knowledge and skills. In manufacturing, computer simulations are being used for many different purposes. Therefore, often a simulation-model might already indirectly be available for optimisation. For example, for simulating control systems, part of the actual control code of Programmable Logic Controllers and/or robots can be reused in the simulation model.

In the work presented in this paper, all robots in the press line are optimised and the objective includes the minimisation of both the cycle time and jerk of the motions. A real-world problem with sheet metal press lines, used in the automotive industry to manufacture the body panels of cars is considered. Reducing the cycle time results in a higher production rate. High jerks makes the motion not smooth and will lead to excessive wear of robots and other equipment. Consequently, this leads to more frequent breakdowns and production interruptions for maintenance interventions. Wear can be reduced by having smooth motions and avoiding abrupt changes in torque because this causes stress and vibrations in the robot structure (Macfarlane and Croft 2003, Gasparetto and Zanotto 2008, Jeon 1995). It was found by Gasparetto and Zanotto (2008) that minimising the integral of the squared jerk guarantees a smooth motion.

The parameters that are optimised are the velocity, path and start/stop signals of the robot motions. In previous work on optimisation of the control of sheet metal press lines, the focus was on just increasing the production rate (Jimenez et al. 2013, Pavel et al. 2013, Glorieux et al. 2014b(a)). Other work also focussed on obtaining smooth motions, but then just for a single isolated press station (Svensson et al. 2013). The goal of this paper is to investigate the effect of including the robot motion smoothness, next to the production rate, in simulation-based optimisation of a sheet metal press line.

The main contribution of this paper is illustrating the importance of including jerk-minimisation in the objective function when optimising the control of press tending robots. With the simulation-based optimisation method used in this work, the
press tending robots of all stations are simulated and optimised together, thereby also optimising the interactions between them. Hence, not only the stations’ interior operations will be optimal but also how it interacts with other stations. First, the potential risk in only considering the production rate, and ignoring the jerk are demonstrated. Second, the obtained solution with the optimisation is compared with a solution created by manual tuning by an experienced operator. The results show that the optimisation can achieve an jerk reduction of up to 35% compared to manual tuning, while maintaining the same overall production rate with the press line.

SIMULATION-BASED OPTIMISATION

Simulation-based Optimisation (SBO), or sometimes just called Simulation-Optimisation, involves computer simulations together with an optimiser to solve an optimisation problem. The role of the computer simulations is to calculate the fitness of the trial solutions generated by the optimiser (Andradttir 1998). The main difference between SBO and ordinary optimisation is that the former problem is solved by computer simulations instead of a set of mathematical expressions. This becomes necessary for real-world, large-scale, complex optimisation problems ((Andradttir 1998, Pasupathy and Henderson 2011)).

The optimisation problem can be formulated as following

$$\min_{x \in S} f(x)$$

1

with the solution vector $x \in S \subseteq \mathbb{R}^D$, and $S$ is the search space bounded by bound constraints that specify the range for each of the $D$ dimensions (i.e., number of parameters). The solution vector $x = [x_1, x_2, \ldots, x_D]$ contains all $D$ parameters that are optimised. The objective function $f(x)$ represents the computer simulation. Additionally, the simulation also includes the constraints of the problem, as these are usually not known explicitly. The concept of SBO is illustrated in Fig. 1.

A drawback with SBO is that the evaluation of trial solutions may become computationally expensive, depending on the simulation model, and thereby time-consuming. The used optimisation algorithm should only require a limited number of evaluations to find an optimal solution and still handle the complexity (Andradttir 1998). A overview of different modelling an optimisation strategies for such computationally-expensive and large-scale problems is given by Shan and Wang (2010). The strategy that is adopted in this work is to decompose the problem into multiple smaller subproblems that are less expensive to compute and have a smaller number of dimensions. Those subproblems are therefore easier to solve, although one must take special care of the interactions between those subproblems (Omidvar et al. 2014; 2015). Fu et al. (2005) provide an overview of the main approaches and the theoretical developments on SBO up to the year 2005. Andradttir (2006) provides a broad overview of the use of random search techniques for SBO. There, it is shown that a guaranteed global convergence is not difficult to achieve. Pasupathy and Henderson (2011) presented a library of SBO test problems to be able to compare different optimisation algorithms’ performance, which includes integer-ordered, continuous, and categorical variable problems, including different types of constraints (e.g. deterministic and stochastic constraints).

PRESS LINE SIMULATION

A real-world problem of sheet metal press lines, used in the automotive industry to manufacture the body panels of cars, is considered in this work. It typically exists of four to six presses and is a fully automated system with several control systems for the press, the press tending robots, safety signals, etc. There is a separate control system for each press station that can communicate with the control systems of the other press stations.

The control functions of a press station are parametrised in order to be able to adjust/tune. Thereby, it can be reused in each press station and when manufacturing different products. Those parameters must then be tuned for each press station and for each specific product. Tuning the parameters in the press stations’ control system are the focus of the presented work in this paper. The production is usually batch run, so that one single type of product is produced at any time. The press tools, grippers, intermediate fixtures, etc. must be changed to produce a different type of product.

Sheet Metal Press Line

A press station includes a press and the downstream robot (i.e., the robot that loads the products into the press). Each press station has its own control system taking care of that part of the press line. The interactions between the press stations are handled by communications between the control systems. For each product, it is necessary to have specific control parameters per station.

The motions of the robots are divided into different segments. Each segment is dedicated for a specific operation such as unloading plates, loading plates, moving between presses, etc. A robot receives a specific start signal from another press or robot in the press line to start the motion of a specific segment. There is also a similar start signal for closing and
opening the presses. Communicating these start signals synchronises the operations and interactions between the press stations.

The control parameters specify robot velocities, robot paths and start signals for segments of the robot motions and the closing/opening of the presses. These must be tuned specifically for each station because the shape of the dies, grippers and plates varies and collisions must obviously be avoided at all times. Furthermore, the production rate of the line is also highly affected by these parameters. Badly tuned control parameters will result in a lower production rate and can also lead to excessive wear of the equipment. It is thus very rewarding for the industry to optimise these control parameters.

In Fig. 2, a drawing of the considered press line is shown, there the products traverse through the line from left to right (i.e. stream of the line). In the considered press line in this work, a specialised 2D belt robot is used. An illustration of this 2D belt robot is shown in Figure 3. This robot is placed between consecutive presses and is responsible for unloading the downstream press and loading the upstream press. The plates are placed on an intermediate table before they are moved into the upstream press in the next cycle. The fixtures used on these tables can also reorient the products if necessary.

The tool mounted on the 2D belt robot has 2 grippers, one on each side in the direction of the stream of the press line, as shown in Figure 3. In this way, it can pick up or place 2 products at the same time. It picks up the pressed product from the downstream press and the product from the intermediate table. Next, the robot moves to the upstream press. There, it places one product in the upstream press while at the same time it places the other product on the intermediate table. This enables the robot to unload the downstream press and load the upstream press in a single motion. Due to this, there are strong interactions between the different press stations. Hence, the synchronisation of the operations is essential to have a collision-free and high production rate.

**Press Line Simulation Model**

For complex large-scale optimisation problems, like the press line problem, it becomes necessary to use simulation models instead of abstract mathematical formulations (Svensson et al. 2005, Andradóttir 1998). The press line simulation model used in this work was developed in the general language/environment MATLAB. When constructing the simulation model, the necessary data and information was retrieved from the equipments’ technical drawings, specifications, parameter files, PLCs’ control code, measurements, etc.

In the press line model, the main focus is on simulating the motions of the presses and robots in the press line. The press line model also includes the communications of the synchronisation signals between the different press stations. The same tuning parameters that are usually manually tuned by the operators are also available in the press line model. These control parameters include start signals, robot velocities and path locations. During an optimisation, the optimiser sets these control parameters before running the simulation. During a simulation, the production rate and the jerk of all robot motions is calculated. At the end of the simulation, the calculated production rate and/or jerk are returned to the optimiser as evaluation values.
The press line model includes the control of the start signals for synchronising the motions over different segments of the trajectory of the robots and the start signal for the closing/opening of the presses. Therefore, the press line model also includes the control logic for this. The communication of these start signals are crucial for the press line to synchronise the press tending robots and presses and thereby avoid collisions.

The press line model also includes the velocity control algorithm and logic for the robots and presses. This controls the velocity during different segments of the trajectory of the robots and presses.

A path planning algorithm for the press tending robots’ motions is also included in the press line model. This uses the parameters for the robot path locations as input and calculates the specific trajectories for the robots.

The implementation of all these control functions and logic (path planning, velocity, start signals) in the simulation model is the same as in the real press line and are parametrised in exactly the same way. This is obviously necessary to be able to tune these parameters and to have a realistic simulation model of the press line. In this work, only a subset of all parameters of the control functions are optimised. This subset includes the velocity parameters, robot locations and the start/stop signals for the robots of each press station. The other parameters, that are not optimised, are set to the value retrieved from the parameter file used by the considered press line and considered product.

The press line model also includes a collision detection feature to check for collisions between robots and the presses. When a collision is detected, the simulation is aborted. In case of collisions, a penalty is returned to the optimiser as evaluation value. Due to that the robots only move in 2 dimensions, a faster collision detection proposed by Nia et al. (2011) is used in this work. The method is based on a transformation of the 3D CAD model representation into 2D collision curves for the robots and presses. This transformation from 3D into 2D reduces the computation time to check for collisions significantly and thereby reducing the simulation time.

A simulation includes first several “warm-up” strokes of the presses to fill the line with plates. After these warm-up strokes, the model simulates a single stroke of all presses in the line. During this single stroke, the press line model checks for collisions, calculates the production rate and the jerk for all robots in the press line.

The press line model was verified and validated. The press line model’s output was compared with measurements from the real press lines, specifically the robot paths, velocities and accelerations of the robots and presses. The measurements for this on the press line where performed using a laser tracker. The difference between the press line model’s output and measurements (< 100 mm) was permissible for the intended purpose of the press line model.

TIME AND JERK OPTIMISATION

The first, most obvious, objective when optimising the control of a sheet metal press line is to minimise the cycle time, as in the previous work on press line optimisation (Svensson et al. 2013, Glorieux et al. 2015). A low cycle time results in a high production rate. As sheet metal press lines are very costly, both in terms of investment and operation, the industry desires a high production rate. On the other hand, it is also very important to avoid production interruptions due to breakage of the equipment. Therefore, a second objective is to minimise the jerk of the robot motions to increase the smoothness. Smooth robot motions guarantee no abrupt torque variations, and thereby reduce stresses and vibrations in the robot’s structure (Macfarlane and Croft 2003, Gasparetto and Zanotto 2008, Jeon 1995). Thus avoiding excessive wear of the robot’s structure, motors, belts, joints, etc. Smooth motions can, according to Gasparetto and Zanotto (2008), indirectly be obtained by minimising the integral of the squared jerk along the trajectory, during one cycle. For a robot with I joints, this is calculated as follows

\[ J(x) = \sum_{i=1}^{I} \left( \int_{0}^{t_{c}} \left( \dot{\theta}_i(x) \right)^2 dt \right) \]

where \(\dot{\theta}_i(x)\) is the angular velocity in the \(i^{th}\) joint with solution vector \(x\) and \(t_c\) is the cycle time. The jerk is obtained by calculating the second derivative of \(\dot{\theta}_i\). The model calculates the jerk along the trajectory of the robots in the press line during the simulation. The model returns the calculated jerk, together with the cycle time, to the optimiser as the evaluation value for respectively the smoothness and the production rate.

The simulation model is built so that it inherently respects the maximum velocity and acceleration of the robots in the press lines. Specifically for material handling operations such as press tending, the maximum acceleration must be respected to prevent that the plate’s inertia forces exceed the grip forces, and consequently dropping the part. Specifically with assistive grippers (i.e. suction cups or magnetic gripper) and sometimes also for impact grippers (i.e. claws or jaws) this becomes essential.

Typically, real-world industrial applications have a predefined target production rate \(p_{\text{target}}\). Hence, the objective function must be formulated to meet this \(p_{\text{target}}\). Additionally, the objective function needs an estimation of the smoothness of the motions of each robot in each press station. Using (2), the objective function to minimise the jerk (and thereby maximise smoothness) for the target production rate for \(S\) stations, and \(R_i\) robots in each station \(s\), becomes

\[ f_{\text{jerk}}(x) = g(x) + \sum_{s=1}^{S} \sum_{i=1}^{R_s} J_{s,r}(x) \]

where \(J_{s,r}(x)\) is the integral of the squared jerk for the \(r^{th}\) robot in Station \(s\) for solution \(x\) according to (2) and where \(g(x)\) checks whether \(p_{\text{target}}\) is met for \(x\). The function \(g(x)\) is
given by
\[ g(x) = C \left( p(x) - p_{\text{target}} \right)^2 \] (4)
where \( C \) is a scaling constant, \( p(x) \) is the production rate for solution \( x \) and \( p_{\text{target}} \) is the target production rate. In this way, small deviations from the target production rate are not penalised heavily. When the difference becomes larger, the penalty increases rapidly. The constant \( C \) was in the tests tuned so that a difference of ±0.5 strokes per minute to \( p_{\text{target}} \) cannot be compensated by jerk reductions.

**OPTIMISER**

The Constructive Cooperative Coevolutionary (CC) optimisation algorithm, proposed by Glorieux et al. (2014a,b; 2015), is an iterative algorithm. It requires that the problem is decomposed into subproblems that can be optimised separately. To optimise these subproblems, an embedded optimisation algorithm is employed in CC. A well-suited algorithm for the subproblems at hand must be selected. Based on previous work with CC by Glorieux et al. (2015), the Differential Evolution algorithm with self-adaptation proposed by Brest et al. (2010) is used. Because the Differential Evolution algorithm is stochastic, CC becomes also stochastic.

The main rationale of CC is that optimising smaller, easier-to-solve subproblems separately instead of the entire large-scale complex problem at once leads to good optimal solutions quicker. The press line problem is decomposed so that each subproblem represents a press station. A detailed description of the CC algorithm is given by Glorieux et al. (2015), Glorieux (2015).

It has been shown to be a competitive optimisation algorithm compared to other metaheuristics (Glorieux et al. 2014a). CC has also been evaluated for optimisation of interacting production stations (Glorieux et al. 2014a; 2015). The results showed that CC outperforms other existing algorithms, such as Differential Evolution and the Particle Swarm Optimiser.

**IMPLEMENTATION**

Tests are done in this work to evaluate the proposed objective function \( f_{\text{jerk}} \). The results are then compared with an objective function that only considers the production rate, i.e. \( f_{\text{base}}(x) = g(x) \). The jerk with the optimal solution found with \( f_{\text{base}} \) is used as reference for the comparison with the proposed objective function \( f_{\text{jerk}} \). Tests were done with 3 different target production rates \( p_{\text{target}} = 9, 10 \) and 12 strokes per minute. This is to check the influence of different target production rate on these jerk reductions.

The operations of press tending robots in the real press line were measured. The control parameters of the measured press line were tuned manually by the operators, and will be further referred to as objective function \( f_{\text{operator}} \). The feedback available to an operator is the line’s production rate, which is displayed on the operator panel. Additionally, they judge the robot motions (e.g. smoothness) by sight, based on experience and intuition. The measured production rate was 9 strokes per minute. The measurements include all motions of the robots (and the presses). The jerk with the manually tuned solution could thus be calculated using (2) and compared with the optimisation results.

There are 4 press stations in the considered sheet metal press line in this work. In total, 19 control parameters per press station are optimised, 6 are start/stop signals of the robot’s and press’ motions, 5 are velocities for the robot motions, and 8 are robot path locations. Thus, the total number of parameters is 76 (i.e. \( D = 76 \)). Each test has been repeated 30 times to obtain a reliable mean result and standard deviation. This is due to the stochastic characteristic of the CC algorithm. For the tests performed in this work, the termination criterion for the optimisation with CC is 10,000 evaluations, which thus corresponds to 10,000 simulation runs.

**RESULTS**

**Time and Jerk Optimisation**

In Table 1, the results of the tests are shown. The table gives both the mean result values and the standard deviation of the 30 independent repetitions. The result values are the total squared jerk along the trajectory, \( J(x) \), calculated with (2) of the final optimal solution found with the objective functions. The results with objective function \( f_{\text{base}} \) are used as a reference for the comparison. Therefore, these are set to 100 and the results with \( f_{\text{jerk}} \) and \( f_{\text{operator}} \) are then expressed relative to this.

A two-sample Student \( t \)-test is used to determine the statistical difference between the results. The \( p \)-value’s threshold for rejecting the null-hypothesis is a significance level \( \alpha = 0.05 \). This showed that the production rate results are equal to \( p_{\text{target}} \) for all objective functions. All results with the different objective functions \( f_{\text{jerk}}, f_{\text{base}}, f_{\text{operator}} \) are significantly different.

When comparing the results in Table 1, it can be seen that with objective function \( f_{\text{jerk}} \), there is a very significant reduction in total squared jerk compared to the reference test with \( f_{\text{base}} \). The results show that the total squared jerk along the trajectory with \( f_{\text{jerk}} \) is around 12% to 30% of the total squared jerk with \( f_{\text{base}} \), depending on \( p_{\text{target}} \). This indicates that it is highly important to include jerk-minimisation in the objective function for press line optimisation to avoid a high increase in jerk. By not having these high jerks along the trajectory, excessive wear of the robot is avoided and consequently maintenance will be required less frequent.

It can also be seen in Table 1 that the standard deviation of \( J(x) \) is much longer with objective function \( f_{\text{jerk}} \), especially for \( p_{\text{target}} = 9 \) and \( p_{\text{target}} = 10 \), compared to with \( f_{\text{base}} \). When \( J(x) \) is included in the objective function, it consequently becomes predictable, which is obviously also desirable for the industry.

In Table 1, it can be seen that the magnitude of the reduction in \( J(x) \) is different with each \( p_{\text{target}} \). A possible explanation for this is that is due to other control parameters that were not included in the optimisation. Not all parameters of the press
line were optimised in this work. Though, further investigation of the specific relation between $p_{\text{target}}$ and the possible reduction in $J(x)$.

### Comparison with Manual Tuning

As mentioned earlier, the motions of a manually tuned solution for the press line has been measured in this work. The smoothness result of the manually tuned solution is also shown in Table 1, i.e. the column with $f_{\text{operator}}$ for $p_{\text{target}} = 9$ strokes per minute. The production rate and total squared jerk ($J(x)$) are calculated for the measured manually tuned solution in the same way as for the optimisation results. The total squared jerk is also presented relative to the result with $f_{\text{base}}$ and $p_{\text{target}} = 9$.

Furthermore, in Figure 4, the instantaneous jerk with the different objective functions ($f_{\text{base}}$, $f_{\text{jerk}}$, $f_{\text{operator}}$) is plotted against cycle-time for $p_{\text{target}} = 9$ strokes per minute. All three jerk curve plotted in Figure 4 are of the same press tending robot, i.e. the robot from the second press station in the press line.

The result with $f_{\text{operator}}$ and $p_{\text{target}} = 9$ show that an operator also reduces the jerk and thereby obtains smooth robot motions with manual tuning. The total squared jerk with $f_{\text{operator}}$ is just 65% of the total squared jerk with $f_{\text{base}}$. In Figure 4, it can be seen that the jerk-curve for $f_{\text{operator}}$ is lower compared to $f_{\text{base}}$. Especially the peaks in the jerk-curve for $f_{\text{operator}}$ are lower compared to $f_{\text{base}}$, which means that there are fewer sudden changes in acceleration during the robot’s motion. The operators manage to judge the smoothness based on the visual feedback from the motions of the robots, experience and intuition.

When comparing the results with $f_{\text{operator}}$ and $f_{\text{jerk}}$, it can be seen that the optimisation (i.e. $f_{\text{jerk}}$) still outperforms the operators (i.e. $f_{\text{operator}}$). The optimised solutions with $f_{\text{jerk}}$ have a lower total squared jerk compared to the manually tuned solution with $f_{\text{operator}}$. The optimisation with $f_{\text{jerk}}$ results in just 30% in total squared jerk compared to $f_{\text{base}}$ whereas the operators ($f_{\text{operator}}$) achieve a reduction to 65% of the total squared jerk of $f_{\text{base}}$ by manual tuning.

The jerk-curve of $f_{\text{jerk}}$ in Figure 4 is lower compared to the jerk-curve of $f_{\text{operator}}$ and thereby also lower compared to the jerk-curve of $f_{\text{base}}$. The peaks of $f_{\text{jerk}}$ are lower, thus there are again fewer sudden changes in acceleration during the robot’s motion, which in turn results in less wear of the robot.

### CONCLUSIONS AND FUTURE WORK

In this paper, the simulation-based optimisation of the control of press tending robots in a sheet metal press line to improve the production rate and smoothness of the motions is presented. Press lines are real-world large-scale complex systems. It therefore becomes impractical to formulate/use an abstract mathematical representation in a generic way.

The robotic press tending operations of all press stations are simulated and optimised together with the presses’ operations. In this way, the stations’ interactions are also optimised, which would not be the case if the press stations are optimised separately. The control parameters of the press tending robots that are being optimised include the robot velocities, start signals for the robots’ and presses’ operations, and robot path locations.

Smaller robot motions have less abrupt changes in torque and less vibrations in the robot structure and thereby result in less wear of the robots and other equipment. During the optimisation, the square of the jerk along the trajectory is used as

---

Table 1: Mean $J(x)$, calculated with (2), of the optimal solutions of 30 independent repetitions with different objective functions, $f_{\text{jerk}}$ and $f_{\text{base}}$, and manual tuning by operators on real press line $f_{\text{operator}}$

<table>
<thead>
<tr>
<th>$p_{\text{target}}$</th>
<th>$f_{\text{base}}$</th>
<th>$f_{\text{jerk}}$</th>
<th>$f_{\text{operator}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mean</td>
<td>std</td>
<td>mean</td>
</tr>
<tr>
<td>12</td>
<td>100.0</td>
<td>33.1</td>
<td>19.3</td>
</tr>
<tr>
<td>10</td>
<td>100.0</td>
<td>48.6</td>
<td>12.1</td>
</tr>
<tr>
<td>9</td>
<td>100.0</td>
<td>34.5</td>
<td>30.6</td>
</tr>
</tbody>
</table>

Figure 4: The resulting jerk curve of the motion of the press tending robot; comparison of the solutions obtained with different objective functions ($f_{\text{base}}$ and $f_{\text{jerk}}$) and from the manually tuned solution by the operators ($f_{\text{operator}}$).
indicator for the smoothness of the robot motions. This indicator is included in the objective function of the optimisation, together with the production rate. The resulting smoothness of the optimal solution with this objective function is investigated. This is done by comparing it with the smoothness of the optimal solution found with an objective function that only considers the production rate.

The results from the performed comparison show that the robot motions are significantly smoother with this objective function. The smoothness of the motions of the press tending robot is improved considerably when it is considered in the objective function. The squared jerk along the trajectory is reduced significantly (up to just 12%) compared to when the smoothness is not considered in the objective function. This clearly illustrates the importance of including jerk minimisation in the objective function to ensure smooth motions for the press tending robots.

The optimisation results were also compared to a manually tuned solution by skilled and experienced operators on the real sheet metal press line. Compared to the manually tuned solution, the optimisation found solutions that have the same production rate. Furthermore, the optimised robot motions are also more smooth compared to manual tuning. The total squared jerk along the trajectory is significantly lower with the optimised solution. It can be concluded that using simulation-based optimisation leads to smoother robot motions compared to manual tuning.

Manual tuning is performed online, on the press line, and thereby it requires that the production is interrupted. The presented simulation-based optimisation of press lines in the paper is performed offline and it thereby does not interrupt the production. Hence, this will result in significant time-savings for the industry.

Future work should include a detailed investigation of the trade-off relation between the production rate and smoothness objectives. This is necessary to learn how much the total squared jerk can be reduced for a specific production rate. Furthermore, it should be analysed which control parameters have the most influence on smoothness. This should be done for example by using a multi-objective optimisation algorithm that obtains a pareto-front of these two objectives, such as proposed by Zamuda et al. (2007) or Zhao et al. (2014). Additionally, a third objective, i.e. the energy consumption of the material handling robots, should also be added to the multi-objective optimisation of sheet metal press line in future work.

Acknowledgements

This work was preformed at University West’s Production Technology West research centre in Trollhättan, Sweden and supported by Västra Götalandsregionen, in Sweden, under the grant RUN 612-0974-13, PROSAM. The authors want to gratefully thank Nina K. Nia and Volvo Cars Corporation for their support. Also, thanks to Dennis Pfau for his contributions to this work.

REFERENCES


FLEXIBLE DATA LOGGING, MANAGEMENT AND ANALYSIS OF SIMULATION RESULTS OF COMPLEX SYSTEMS FOR EROBOTICS APPLICATIONS

Linus Atorf, Torben Cichon, Juergen Rossmann
Institute for Man-Machine-Interaction
RWTH Aachen University
Ahornstrasse 55, 52074 Aachen, Germany
E-mail: atorf@mmi.rwth-aachen.de

KEYWORDS
3D Simulation, Simulation Database, Data Logging, SQL, eRobotics, Multi-Body Dynamics

ABSTRACT

The general idea of simulation is the imitation of a real-world process and thereby the prediction of the dynamic behavior of complex systems in a time- and cost-efficient manner. Possible applications of this technology are endless, yet further storage and management of the simulation results is still under discussion. Comprehensive simulation capabilities are already today the method of choice to cope with the complexity, cost, and safety issues from simple physical tests to advanced robotic applications in highly dynamic environments. The tool chain of simulation often ends with a “final” result or a specific finding. To extend, promote, and enhance simulation capabilities, streamlining and simplification of the logging, management, and analysis procedure of simulation is the central aspect of this paper. We show a flexible, efficient, and easy to use mechanism to store simulation data in SQL databases, capable of handling a huge amount of data, in order to develop a standardized management and evaluation process for analysis of simulation results.

INTRODUCTION

Computer simulations are ubiquitous in today’s world of engineering—especially within the emerging field of eRobotics—and the number of professional simulation systems is continuously increasing. While 3D simulations can produce powerful, amazing, and convincing renderings of virtual worlds, Virtual Testbeds in the field of eRobotics [Atorf et al. (2013)] can also be used to test and evaluate real world behavior within the simulation. The reasons we simulate and the advantages of simulation are the riskless, cost-efficient, and repeatable execution of scientific experiments with the option of multiple simulation cycles to finally visualize and realize ideas in a reproducible, comprehensible, and scientific manner. Nevertheless, the central, most significant aspect of using simulation is still to understand complex models in detail. Consequently, there exists a growing need for thorough numerical analysis of simulation results.

DATABASES, SIMULATION, AND SQL

Using simulation results for prediction purposes requests a potent data storage, management, and analysis system behind the simulation tool itself. To motivate the idea of this paper we want to start with the main idea of simulation: understanding a given setup and evaluating its potential.

Figure 1: Dynamical Multi-Body Simulation of a Truck: Starting Point and Temporal Trajectory-Overlay for Different Steering Angles

In figure 1 one can see a 3D simulation series of a truck driving towards a ramp with regards to different steering angles. In general, everyone would expect a continuous spreading from left to right. At the end of this paper, we will show that simulation alone is not capable of showing all details, and consequently its full potential, directly. To be able to analyze anything produced by a simulation it is important to incorporate an adequate logging procedure. This ranges from automatically logging everything and enabling the user to post-process results in an encapsulated environment (such as for example in MSC.Adams [Cichon et al. (2014)]) towards defining specific features to be logged, exported, and evaluated with external programs. Additionally, the format of ex-
port or data storage has to be defined and standardized, enabling reproducible evaluation processes and streamlining of evaluation procedures.

In the following, the main idea of logging and an introduction to data storage and databases is given, concluding with the state of the art in today’s data logging, management, and analysis systems for and within simulation tools.

**Main Idea of Logging**

Analyzing simulation results is important and repetitive—a step that always has to be carried out. Many tools (in fact: a whole market of tools and a world of methods) exist, from visual programming (LabView, “WYSIWYG”) to scripting and programming languages (e.g. Python with NumPy, SciPy, and matplotlib), from office software (Excel) to advanced numerical engineering packages (Mathematica, Maple, MATLAB, etc.). Very often, both simulation software as well as analysis programs, whether custom-written or run via standard software, are heavily customized.

A common unified data format is often not possible or is too entangled and dependent on the area of research. This leads to a lot of repetitive, low-level, and boilerplate development effort: How to get (numerical) data from simulation runs into the analysis module. If simulation and analysis software is combined into one single package, the following questions arise: How can data be exported for further evaluation in different software packages? Can all (relevant) data be saved, or only a subset? It is always possible that certain data types are missing in early versions of computer simulation models. How can they be added to the existing numerical results? Does it make sense to have an incorporated (proprietary) data analysis tool or does it make more sense to use available, much more powerful, analysis tools and “only” develop a commonly used interface—in form of a universally accepted database format—to evaluate simulation results in “private” and in general?

According to good scientific standards, results of experiments (whether “real” physical experiments or simulated ones) have to be stored and made accessible to whoever asks. The data format of simulation results is, of course, not standardized and it probably cannot be, as the areas of applications are too manifold. Nevertheless, using CSV-formatted (comma separated values) text files has become one standardized format to store, exchange, and evaluate data sets. Still, this format has its own problems. Starting with the discussion of comma versus period for decimal places or the thousands separator, the string based CSV data import often has encoding problems (UTF-8 versus ASCII) or ambiguous escape rules. Furthermore, there is no standardized header format for CSV files and frequently import problems for data sets with irregular number of columns occur. Though, it will always be popular and has its right to exist because almost any software or editor can read it. So maybe the CSV-format is one of the “greatest common” features, which any software solution to the data logging problem has to provide at least as export method. But on top of this generally accepted common denominator for data storage and analysis, there has to be a way to overcome the aforementioned drawbacks of CSV becoming the de-facto standard for simulation data storage and as evaluation back-end.

In this paper, we propose a solution for many of the stated problems. Our suggestion is to store all relevant data into an especially designed SQL-database. We present a possible database scheme and methods to store data efficiently and performance-oriented. We continue to show how data analysis (of simulation results) can be carried out conveniently by using SQL-queries. Furthermore, the database scheme and a suggested “basic set” queries provide a foundation for standardized access to simulation data via the well-known and worldwide-popular SQL-interface ([Date and Darwen (1993)], [Hipp et al. (2015)]).

Many engineers, scientists or developers in general fear for escalating complexity, development effort, and performance penalties when thinking about evaluating databases or database connections for their product, and favor a seemingly simpler solution—mostly the aforementioned CSV export. We will show that this fear is unsubstantiated in many use-cases and that the benefits of using SQL-queries on databases usually outweigh the additional development efforts.

**Introduction to SQL databases**

Within the last decades, SQL has become an almost definitive standard for databases. Via SQL, one has (almost) full compatibility among the databases to use and there is no vendor lock-in. This lead to millions of SQL databases around the world. SQLite ([Hipp et al. (2015)]) is one stand-alone SQL database engine with almost no configuration-overhead. It is the most widely deployed database engine in the world, probably due to its speed and simplicity, based on a single, serverless database file without authentication. It is free and open source, interfacing with almost all available programming languages. Furthermore, SQLite is already included in many frameworks as default (e.g. Qt, Python, etc.), due to its stability (millions of unit tests with 100% code coverage), popularity (foundation for e.g. the Android OS and many common web browsers), as well as a long term support perspective.

**State of the Art**

In this section the introductory part of simulation databases and using SQL databases for this purpose will be connected to the current state of the art in building
and/or using databases for simulation. Firstly, some approaches of simulation database developments and usages are presented before we shortly present the differences between SQL and NoSQL.

As we have seen in the introduction, the use of a database back-end within a simulation system is very promising. The realization of such databases for simulations are diverse in their setup, general idea, and their use. In [Riebe (2012)] or [Malinowski and Suchy (2010)] for example, the general workflow of modeling, simulation, data storage, data analysis, comparing results with observations, and finally sharing and publishing the results is presented, to show the promises of an easy accessible database for reproducing and visualizing results by using for example web interfaces. This shows the SQL-based approach of a generally accepted, standardized interface to take advantage of, and use it for your data. Furthermore this shows the streamlining capabilities using such databases. Another approach of simulation databases and queries is presented by [Oberweis and Sänger (1994)] who developed a graphical query language for temporal data models for easier accessibility using petri nets. This are all examples of using simulation databases. Based on our introduction, we will focus on the standardized SQL database back-end for simulation to harness the potency of SQL.

As the counterpart to SQL, so called “NoSQL” has been established as a wording for alternative databases in general. “Not only SQL” are non-relational databases not using the SQL API, and being famous for horizontal scalability and being schema-free. In the following, some differences will be shown between SQL and NoSQL always focusing on the strength of each database model.

In general, SQL is the standard for most known databases and used by almost everyone in the past—the “one size fits all” solution. There are manifold plugins and extensions to support and simplify the use of such databases. The major difference to NoSQL is the fact that SQL databases have a unified, powerful query language—SQL. Therefore, it is obvious to use SQL databases when possible.

However, if the standardized query language is not as important compared to very requirements on performance and scalability, one can evaluate using a NoSQL database, which can for example be based on graphs, which would in turn enable the function of cross-linked information storage. For more information about NoSQL see e.g. [Leavitt (2010)] or [Cattell (2011)].

Taking everything into consideration, an SQL database—especially using the SQLite engine—is a promising way to bring an fast, efficient, easy, standardized, and commonly accepted data management system into simulation.

**ENTITIES IN SIMULATION SYSTEMS**

Simulation systems usually work with some sort of “entities” (or objects) that are part of the models they simulate: Rigid bodies, machines, controllers, particles, physical devices, equations. These entities are very similar to the object-oriented programming paradigm. Entities are instances of a certain class. Entities have attributes; attributes have data types. For example: In a dynamic rigid body simulation, there might be a rigid cube. During simulation, it falls down due to gravitational acceleration (*Falling Box Example*). Let the cube be an entity named “MyBox” of the class “RigidBodyCube”, and let it have attributes such as edge length, weight, 3D position, 3D orientation. The data types of these attributes could be, respectively: double precision floating number (in short: double), another double, 3D vector of doubles, $3 \times 3$ rotation matrix (9 doubles). During simulation, some of these attributes may stay constant, and others may change. For example: due to gravity, the position may change, and in our example only the z-component of the 3D position vector would vary. For our logging-approach of simulation results, we assume that all relevant variables can be represented in this fashion, i.e. all quantities are attributes of certain data types, belonging to entities of certain classes. The simulation system does not necessarily have to be programmed or designed object-oriented, but the data variables to log should at least be classifiable to this system. The easiest way would be to simply use variable name for identification.

For further discussion, we rely on the following notation, used e.g. by the VSD-database of the VEROSIM simulation framework [Roßmann et al. (2013)]. Here, entities are called *instances* of a certain class. Classes can have attributes, which we call *meta-properties*. This is in fact the concept of properties of the popular *Qt* or *.net* frameworks, or the Java programming language. During runtime, an instantiated meta-property always belongs to exactly one instance, hence we use the term *instance-property*. Only instance-properties have actual values (in short: prop-values); meta-properties describe their name and data type. Continuing the aforementioned *Falling Box Example*, we have an instance “MyBox” of the class “RigidBodyCube”. The class has the meta-property “weight”, which becomes an instance-property (with the same name “weight”) of the instance “MyBox”. We summarize the definitions and assumptions for further discussion:

- Instances have a name, e.g. “MyBox”, or “” (empty)
- Instances have a class, e.g. “RigidBodyCube”
- Instances have a unique “instanceID” (integer), e.g. their memory address or some hash
• Classes can have meta-properties
• Meta-properties of classes become instance-properties of instances
• Meta-properties have a name, e.g. “weight”
• Meta-properties have a data type, e.g. “double”
• Instance-properties have a meta-property; they have the same name and data type as their according meta-property
• Instance-properties have exactly one parent-instance “they belong to”
• Instance-properties have actual values (called prop-values), e.g. “7.5”

LOGGING AND DATABASE BACKEND

The key mechanism of our approach is to log all modifications of all observed instance-properties. Whenever a prop-value (remember: the value of an instance-property) changes, this event is stored in a database together with the simulation time and the wall time since simulation-start.

First of all, the prop-value’s modification must be detected and processed in the simulation system’s programming language. This can usually happen inside the property’s accessors (i.e. the setter-function). Consider the property “weight” and its setter `setWeight(double newWeight)`. If `newWeight` differs from the current prop-value, we have an event to log. We notify the logging-subsystem by either calling a function directly, by invoking the Observer Pattern [Gamma et al. (1994)], or by raising an event or signal, if event-handling such as e.g. Qt’s “signals & slots” feature is present. The data we want to pass to the event-handler are:

• Timestamps: simulation time and optionally wall time
• ID of the current instance-property (“propertyID”)
• New prop-value

The newly introduced `propertyID` (integer) is unique for each instance-property. It can be created by a bijective mapping function \( f \) with \( propertyID = f(instanceID, property-name) \). The `instanceID` of the parent-instance has to be unique for each instance. A simple implementation could use the instance’s memory-address (i.e. the pointer-value), as long as it does not change during simulation. To manage `propertyID`s, a dictionary or hash-table can be used to look up (instanceID, property-name)-tuples time-efficiently and issue fresh, consecutive IDs whenever a new instance-property is seen.

Let the instance “MyBox” from our earlier Falling Box Example have `instanceID = 1234` and another instance of the same class, “OtherBox”, have `instanceID = 6789`. The IDs for their properties “weight” could then be \( propertyID = 1 = f(1234, “weight”) \) and \( propertyID’ = 2 = f(6789, “weight”) \), respectively.

There are several possibilities for storing the prop-value into the database. We opt for serializing values to string representation. This means that all prop-values can be always stored in a column with SQL-data type `VARCHAR` or `TEXT`. While some programming languages provide built-in serialization-functionality (such as e.g. Java), others do not. However, often 3rd-party solutions exist (e.g. Boost.Serialization for C++). Alternatively, text-streaming operators for numeric values can usually be implemented with low effort. A 3D-vector could for example just store its components in string-form as well-defined form of token-separated values without ambiguity.

Using text-format to store prop-values focuses on flexibility for data processing and analyzing. If using serialized string values has a negative impact on execution speed or database size—which cannot be tolerated in some cases—alternative choices for storing prop-vals are: using a BLOB datatype and binary serialization, or providing multiple columns (in separate tables, if applicable) with according data types, such as `INTEGER`, `FLOAT`, and `CHARACTER`.

The database schema we propose is shown in table 1. The table “PropertyModLog” is the core of our concept, where the events we just discussed are stored. The attribute `jobID` is used to distinguish between different simulation runs of the same simulation model, possibly with different parameters or start conditions. The `jobID` is simply an integer that only has to stay constant during one simulation pass, similar to `instanceIDs` and `propertyIDs`. In consequence, `instanceIDs` and `propertyIDs` are not guaranteed to stay constant across multiple simulation runs.

Logging only events of modified prop-values is not enough for flexible and powerful data analysis. The tables “PropertyDetails” and “InstanceDetails” are used as legend and to support comparing prop-value time series from different simulation runs. For every (jobID, `propertyID`)-tupel from `PropertyModLog`, `PropertyDetails` contains name, datatype, and the parents `instanceID` of the instance-property. Given (jobID, `propertyID`) to look up `instanceID` in `PropertyDetails`, we can obtain the (jobID, `instanceID`)-tupel to retrieve further details about a property-instance’s parent, such as its name and class.

Finally, we have two more optional tables: “JobDetails” can be seen as a simple “table of contents” for several simulations runs. “SimTimeLog” can be used for performance measurements. It is designed to contain “timing measurement points” to estimate simulation speed and logging overhead. The idea is to insert rows regularly, for example at every 1ms simulation time. Wall time is measured since the start of the simulation. Utilizing the column `selfTime`, we suggest to time the duration for serializing prop-values.
Table 1: Database Schema (Color-coded for Understanding Conjunctions)

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Datatype</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table JobDetails</td>
<td></td>
</tr>
<tr>
<td>jobID</td>
<td>Integer</td>
</tr>
<tr>
<td>timeSTamp</td>
<td>Text</td>
</tr>
<tr>
<td>comment</td>
<td>Text</td>
</tr>
<tr>
<td>Table InstanceDetails</td>
<td></td>
</tr>
<tr>
<td>jobID</td>
<td>Integer</td>
</tr>
<tr>
<td>instanceID</td>
<td>Integer</td>
</tr>
<tr>
<td>instanceName</td>
<td>Text</td>
</tr>
<tr>
<td>instanceClass</td>
<td>Text</td>
</tr>
<tr>
<td>Table PropertyDetails</td>
<td></td>
</tr>
<tr>
<td>jobID</td>
<td>Integer</td>
</tr>
<tr>
<td>instanceID</td>
<td>Integer</td>
</tr>
<tr>
<td>propertyID</td>
<td>Integer</td>
</tr>
<tr>
<td>propertyName</td>
<td>Text</td>
</tr>
<tr>
<td>propertyType</td>
<td>Text</td>
</tr>
<tr>
<td>Table PropertyModLog</td>
<td></td>
</tr>
<tr>
<td>jobID</td>
<td>Integer</td>
</tr>
<tr>
<td>propertyID</td>
<td>Integer</td>
</tr>
<tr>
<td>simTime</td>
<td>Real</td>
</tr>
<tr>
<td>wallTime</td>
<td>Real</td>
</tr>
<tr>
<td>value</td>
<td>Text</td>
</tr>
<tr>
<td>Table SimTimeLog</td>
<td></td>
</tr>
<tr>
<td>jobID</td>
<td>Integer</td>
</tr>
<tr>
<td>simTime</td>
<td>Real</td>
</tr>
<tr>
<td>wallTime</td>
<td>Real</td>
</tr>
<tr>
<td>selfTime</td>
<td>Real</td>
</tr>
</tbody>
</table>

Continuing the Falling Box Example: Let us assume we have only one simulation run stored, so jobID = 1. When we know the instance-name “MyBox”, we can look up its instanceID = 1234 and that this instance has a property with propertyID = 1. Using a (jobID, propertyID)-lookup in PropertyDetails, we learn that the instance-property is named “weight” and of data type “double”. A time series of its prop-values can be retrieved by filtering PropertyModLog for this same (jobID, propertyID)-tuple.

Note that our database schema does not specify primary keys for tables. This allows for greater flexibility and improves performance: no checks for uniqueness have to be carried out by the database engine when inserting rows. In SQLite, a primary key column “rowid” is introduced automatically in such cases. Data rows should be logged with regular INSERT statements in SQL, preferably wrapped in one big transaction for a whole simulation run for performance reasons. To further increase database performance with e.g. SQLite, pragmas to keep the transaction-log in RAM can be used, or even a whole in-memory database. Using on-the-fly file-system-compression can reduce database size.

We observed a reduction in file size of about 50% when using NTFS on-the-fly file system compression without any measurable performance penalty.

For later data analysis, we recommend creating indices on the columns most often queried. However, these indices should only be created when all data logging is completed to avoid additional computing load on the database during insertions.

DATA ANALYSIS

The database schema we presented in the previous chapter is designed for efficient data logging and flexible, comfortable data analysis. We believe that SQL is ideally suited for processing logged simulation results and illustrate the possibilities with just a few queries. During all times, a jobID must be given. If only one simulation run has been recorded, clauses with jobID can be omitted.

First of all we show how to retrieve time series of given instance-properties. Ideally, a combination of instance-name and property-name is unique, e.g. “MyBox” and “weight”. We then first query InstanceDetails for the instanceID, followed by a lookup of the propertyID from (instanceID, property-name). The propertyID is all we need for queries in PropertyModLog. All steps nested in one single query can be seen in listing 1 (additional WHERE-clauses using jobID are omitted for brevity).

```
SELECT * FROM PropertyModLog WHERE propertyID =
  (SELECT propertyID FROM PropertyDetails WHERE propertyName = 'weights' AND instanceID =
   (SELECT instanceID FROM InstanceDetails WHERE instanceName = 'MyBox'))
```

Listing 1: Exemplary SQL Query for the Falling Box Example (Time Series of a Property)

To compare time series of different instance-properties, one can of course execute several queries after one another and compare the results. A comfortable and convenient way for exploratory data analysis is to let the database return time series of different properties with common time points for easy plotting (see listing 2 and figure 2 illustrating the results of such a query).

```
SELECT * FROM
  (SELECT simTime, value AS v1 FROM PropertyModLog WHERE propertyID = 137) t1
INTERSECT
  (SELECT simTime, value AS v2 FROM PropertyModLog WHERE propertyID = 139) t2
ORDER BY t1.simTime + t2.simTime
```

Listing 2: Exemplary SQL Query to Compare Two Property Time Series

So-called aggregate functions of the SQL standard come in handy for basic statistical analysis of prop-values. The query shown in listing 3 returns number of elements, average, minimum, maximum, and total sum of all values of a scalar instance-property. Additionally, also basic mathematical operations are supported by SQL (note that, depending on the database engine, additional cast-
ing functions to convert strings into numbers may have to be used).

```
SELECT count(value), avg(value), min(value), max(value), std(value)
FROM PropertyModLog WHERE propertyID = 1 AND plotID = 1
```

Listing 3: Exemplary SQL Query to Calculate Basic Statistical Quantities of Property Time Series; See Figure 2 for Visualization

Since the value-column of table PropertyModLog contains serialized string representations, more complex evaluations of results often require de-serializing the strings first. While this is one possibility for further data processing, it is also possible to introduce filter-functions that extract scalar quantities from such strings. For example, one can easily retrieve the x-component of a 3D vector whose elements are stored as string-list, simply by parsing and converting the first string token. Using this approach allows easy access and processing of interesting observables, without full de-serialisation or un-marshalling of stored data objects.

![Figure 2: SQL Query using “DB Browser for SQLite” [Piacentini et al. (2012)]; Result of Listing 2](image)

Most SQL database engines, including SQLite, already provide a toolchain to export CSV files of query results. This ensures backwards compatibility to all other applications, where raw CSV data files are still needed—they can easily be generated from our results database.

APPLICATIONS

To illustrate the usefulness of our concept, we implemented SQL-based logging into the simulation framework VEROsim [Röffmann et al. (2013)]. The database engine SQLite we chose is already included in the Qt programming framework for C++. For evaluation, we implemented convenience functions in MATLAB which execute SQL-queries just as demonstrated in the previous chapter. The example application presented here features a dynamic multi-body simulation of a truck driving over a ramp (compare figure 1). In several simulation runs lasting 7 seconds, we set the steering angle to different fixed values in the range $[-85^\circ, +85^\circ]$ and recorded the results. Due to VEROsim’s powerful active simulation database VSD, which features a meta-type and event-notification system, we included all available instance-properties into the database log. For this example, we observed about 500 instances with about 1300 changing instance-properties per simulation run, including state vectors of the multi-body simulation engine. In total, we recorded just about 8 million events in table PropertyModLog, resulting in more than 100,000 events per second simulation time. One explanation for this surprisingly high event rate could be the 1 millisecond step time of the dynamic simulation engine. The SQLite database was stored on a regular SSD drive; its size is 800 MiB, about 1 GiB with indices included. Using NTFS on-the-fly file system compression on Windows, file sizes are about half that size without deterioration in performance. Roughly 50% of the computing time was used for logging overhead, i.e. the simulation model could be run in half real time on a typical workstation with an Intel Core i7 CPU.

While creating indices took several minutes computing time, typical queries we execute all take less than one second while returning thousands of rows—very often less than 100 milliseconds when cached by the database engine.

For visual analysis of the simulation results, we identified the instance corresponding to the simulated truck’s chassis, which is moved by dynamic multi-body simulation. A suitable instance-property for demonstration is “worldFrame”, which represents the homogenic transformation matrix ($\in \mathbb{R}^{4 \times 4}$), i.e. the pose. Our serialisation format outputs 16 double values, separated by the token “;”. Using a filter function, we could easily extract the 3D translation vector from the matrices. Plotting those different truck positions over time results in the trajectories shown in figure 3. A slight asymmetric behavior is visible due to the ramp, which was placed biased slightly to the right.

As we noticed during this analysis, very steep steering angles do not always have the intended result. With steep steering angles exceeding approximately $\pm 60^\circ$ to the left or right, slippage occurs—the truck drives (or even slides) along curves less narrowly when steering too intense. This behavior can be noticed in the blue lines in figure 3.

CONCLUSION

In this paper we have presented an efficient and flexible way to store simulation results and generic time series in SQL databases for further processing and analysis. The database schema and related queries to store and extract data are the foundation of our concept. They enable powerful data analysis using the widely adopted SQL language as common interface to a wide range of existing tools. We implemented the ideas presented here in the VEROsim simulation framework, where they have
proven useful in prototyping and production scenarios. Performance penalties are often neglectable; even with heavy load on disk and CPU usage, the benefits far outweigh the disadvantages in our experience. We recommend using a database-driven logging mechanism in simulation systems such as the one we proposed. It can be very useful to have access to the history of simulation variables during debugging or post-mortem analysis, just like debuggers use core dumps for application crashes. When determinism in simulation models is desired, a comparison of recorded data logs can be utilized for verification. Another application scenario are automated test cases, where certain conditions of instance-properties can easily be supervised and queries can be used to verify that constraints for certain prop-values are satisfied.

ACKNOWLEDGEMENTS

Parts of this work were developed in the context of the research projects INVIRTES and ViTOS, which are funded by the German Aerospace Center (DLR) with funds provided by the Federal Ministry of Economics and Technology (BMWi) under grant numbers 50 RA 1306 and 50 RA 1304.

REFERENCES

Atorf L.; Kaigom E.G.; and Roßmann J., 2013. A New eRobotics Approach to Simulation-Based Analysis and Optimization. In DeSE 2013, 6th International Con-

Cattell R., 2011. Scalable SQL and NoSQL data stores. ACM SIGMOD Record, 39, no. 4, 12–27.


FISH MOTION SIMULATION

Ievgen Smielik, Klaus Müller and Klaus-Dieter Kuhnert
Institute for Real-Time Learning Systems, University of Siegen
Hölderlinstr. 3, D-57076 Siegen
Germany
ievgen.smielik@uni-siegen.de

KEYWORDS
Estimation, tracking, practical implementation

ABSTRACT

In this paper we describe the retrieval of some real fish properties to establish them on a virtual fish. These include swimming motions. We introduce a special function which extracts a definite description of a fish. We use polynomial interpolation and spline to estimate fish body changes (moves), then reproduce such on a display.

INTRODUCTION

This paper describes work on a project called "Virtual fish". It is a cooperation between biology and computer science departments. The goal of the project is to achieve a virtual fish, that can visually interact with a real fish. We set a display showing a virtual fish next to aquarium with a real one, then alive fish should acknowledge an artificial one and both act accordingly. Fish visualization was implemented already. In short, we use a textured 3-D model with line segments (or bones), that represent fish backbone. Each bone can move, change its angle with another bone that it is connected with. All the bones can move in only one common plane. Each line segment has length, approximated from real fish images. Using our model allows us to draw a virtual fish on a screen to simulate a real fish. The central question of biology research here is a mate-choice, and as there are many aspects, we keep the system as flexible as possible: bone count, fish look can change.

Focus of this paper is on estimation of a fish swimming movement and its application to a virtual model. Described method is independent of fish species. We give details on how to describe and simulate a natural process on a computer.

To study fish behavior we set up a stand with aquarium and top view high speed camera. We record hours of video of fish behavior in aquarium for later analyse.

There are several works on fish simulation, authors of (Kim 2007) are mostly focused on a system performance in visualization, they also use a skeletal fish model.

There are authors using a robotic fish for simulations. Work (Byuh 2013) is focused mainly on a robot tracking. Reference (Tavenna 2010) has a main focus in a water flow around fish body and (Han 2008) has as well hydrodynamics as a basic topic.

Authors of (Terzopoulos 1994) use reinforced learning to create an artificial fish, which is an extensive task, our goal at presented stage is a motion simulation. We use a function approximation to reproduce actions of a real fish on a display.

Multi-joint approach is presented in (Qinyuan Ren 2014), authors combine oscillator with artificial neural networks to generate desired motion patterns.

IMPLEMENTATION

Fish model

We use our model with bones for both real fish tracking and virtual fish representation. On the one hand we map our model and a real fish, and on the other hand we display an achieved model.

Find fish area

To find a region where a fish is located we use segmentation technique described in (Klaus Müller 2014), basically background subtraction. Applied to an image with a fish, it results in a set of pixel coordinates where a fish is located, a fish area.

Find middle curve

With a fish area we have no definite location of a fish and no grip on finding bones, while we have to find the best possible location for our model bones. We transform a plain fish area to a set of points that describe a fish position exactly. As such points we selected a middle curve, we find locations inside a fish area evenly spaced between two fish sides. Fish sides are two parts of a fish contour, we find a contour with approach from (Suzuki and Abe 1985) and divide it in two pieces from head to tail and from tail to head. Location of head or tail we call a vertex on analogy with parabola vertex. First we find an approximation for vertex, to do that we fit the
whole fish area into bounding rectangle and find left-
most and right-most points, these become both vertex
estimates.
We compute a fish contour $C = \{C_k\}, k \in [1, P]$, which
is a sequence of $P$ points, where $C_h$ and $C_t$ are the
nearest to estimated vertexes and $h < t$. $h$ and $t$ indexes
split the fish contour in two parts: from head to tail and
from tail to head, we define

\[
A_{app} \equiv A_{app} \ni C_k, k \in (1, h) \cup (t + 1, P)
\]

and

\[
B_{app} \equiv B_{app} \ni C_k, k \in (h + 1, t).
\]

A fish area contour near a vertex represents a part of
parabola, but fish sides become almost parallel near or
behind side fins. We use that fact to find the precise
vertex location. We note that for some points $P$ from
$A_{app}$ the closest point in $B_{app}$ is the first one. Points $P$
lay on a parabola, but when two side curves go parallel
then the closest point is not the first, let this point index
be $r$. Fig. 1 shows the approach graphically.

Figure 1: Vertex search. White is a fish area, black is
background, red circles mark fish contour, blue lines
connect minimum distant points on different sides,
yellow circle is an approximation for vertex, cyan is a
first point from $A_{app}$.

We apply this method to find parabola end on other
side and get point index $l$. Then real vertex index is
the middle between $l$ and $r$, $v = \frac{l + r}{2}$, point $C_v$. And
real contours $A$ and $B$ achieved. We need to find the
middle between point sets $A$ and $B$, first we filter point
sets, we have to remove fish side fins (which are also in
a fish area). The issue here is that side fins are more or
less transparent, but sometimes with a different angle or
light conditions fins appear in a fish area, and we came
up with a following approach, filtering takes a point from
one side if it is at the minimum distance from some point
from another side,

\[
min A = \{A_k \in A\} : \exists B_m \in B, d(A_k, B_m) <
\]

\[
d(A_t, B_m), A_t \in A
\]

and

\[
min B = \{B_k \in B\} : \exists A_m \in A, d(B_k, A_m) <
\]

\[
d(B_t, A_m), B_t \in B,
\]

we achieve two smaller sets, having eliminated unwanted
points, mostly fish side fins and also minor reflexions,
and then we compute the fish middle curve, it contains
middles between minimum distilled points from $minA
and minB$. Let $middle Curve = \{P_k\}, P_k = \frac{A_m + B_k}{2} :$

\[
A_m \in min A, B_k \in min B, d(A_m, B_k) < d(A_m, B_t), B_t \in
\]

$min B$. The result of finding a middle curve is shown on
Fig. 2.

Figure 2: Fish middle curve

Estimate bones’ angles
Till now we got fish middle curve $\{P_k(x, y), k \in [1, C]\}$,
we divide it into pieces, which lengths correspond to
bones length. Let our bones sizes be $l_m, m \in [1, N]$. We
choose such a set $S_m = \{P_k\}, k \in [r, p], m \in [1, N]$ of
subsets of $\{P\}$, that sum of length between consequent
points from one set, $\|S_m\| = \sum_{i=r}^{p} d(P_i, P_{i+1})$, equals
to one bone size, $\|S_m\| = l_k$ and every bone has such a
set of points. Then we fit a line into each bone’s points.
Line equation is $k * x + b = y$, by judging $k$ and $b$ as
variables and taking $x$ and $y$ from point coordinates, we
get an overdetermined system of linear equations that
we solve in terms of least-squares problem with singular
value decomposition method. As a result we have $k$ and
$b$ line coefficients for each bone. Bones are shown on
Fig. 3.

ESTIMATE BODY MOVING
Now we can track the fish and extract information about
its motion.
Find line trajectories

We search our fish videos for periods where fish moves mainly in one direction. We use our line fitting routine to check if a series of a consequent fish locations (taken from following frames) are nearly at one line. Figure 4 shows one such period of linear swimming, full fish middle curves are plotted. Figure 5 gives a different view.

There are three dimensions for our fish model: axis parallel body length, axis perpendicular to body length (from top view, describes mostly tail displacement) and time. On Figure 4 the time is shown in different colors, two other dimensions are horizontal and vertical. Fish can bend in only one plane, turn a tail from side to side, so we need only two axis to describe fish body state.

We note that any point on a fish tail follows sinusoid, so we use sinusoidal function in time dimension, while polynomial represents fish body bending. That leads to

\[ y(x, t) = P(x) \cdot \sin(B \cdot t), \]

where \( x \in [0, L] \) is body length measure, \( L \) is estimated 200, \( t \) is time (in terms of a frame number), \( P(x) \) is a polynomial function, value of \( B \) affects the period, estimated value is 0.4, function result, \( y \) is a fish body displacement.

Interpolate body curve

We use a general polynomial function equation in form

\[ P(x) = \sum_{i=0}^{p} c_i \cdot x^i, \]

where \( p \) is a polynomial degree. We want to find coefficients for such curve that fits our fish middle curve, we solve a system of linear equations. Number of equations is amount of points in middle curve, \( C \), coefficients are unknown and we take \( x \) and \( y \) values from middle curve points coordinates. In that way

\[
\begin{align*}
\begin{cases}
c_0 + c_1 \cdot x_1 + c_2 \cdot x_1^2 + \cdots + c_p \cdot x_1^p &= y_1 \\
c_0 + c_1 \cdot x_2 + c_2 \cdot x_2^2 + \cdots + c_p \cdot x_2^p &= y_2 \\
\vdots \\
c_0 + c_1 \cdot x_C + c_2 \cdot x_C^2 + \cdots + c_p \cdot x_C^p &= y_C 
\end{cases}
\end{align*}
\]

We solve the system against unknown coefficients and get the best fit approximation to fish middle curve. But we note that not all the middle curves are good representable with a single polynomial, we divide fish movement in two periods in time: first period, when tail moves from zero to a point of the maximum amplitude and second period, when it moves back to zero. Fish starts to move from a zero point, when all bones lay nearly at one line, middle curves at different time at the period are concave. Plot of estimated polynomial is presented on Figure 6. After it reaches the maximum amplitude, middle curve functions become convex, one such plot is given on Figure 7. On the plots, horizontal axis represents fish body length while vertical shows bone displacement. Of course, there is only a finite number of bones, so we fit line segments into a curve and calculate angles between them.
Interpolate between two polynomials

In order to change middle curve representation from one polynomial to another, we first used a third polynomial. Coefficients of a third polynomial are averages of corresponding coefficients of another two polynomials. Figure 8 shows concave polynomial on top, convex at the bottom and an average one in the middle.

If we simply define time frames when to use which polynomial then moving function is non-continuous. Figure 9 shows transition between polynomials with definite time frames. On this plot blue areas show values of a first period polynomial, green areas show values of a second period polynomial, red areas - an average one.

To improve fish movement we change from defining another polynomial to finding mean values. Figure 10 shows transition between polynomials with average values.

We have used interpolation in fish length dimension to approximate middle curves, and now to achieve continuous motion function of body length measure and time, we use spline interpolation in time dimension, Fig. 11.

CONCLUSION

In the presented work we show how to copy a real fish movements for use in computer environment. Behavior of achieved virtual fish is realistic as confirmed by biology expert. Our method base is to find a function that describes an object (or a process) and analyze the function. Major steps of current work include finding a middle curve, approximate it with polynomial function, build spline from values of polynomial. Middle curve of a fish is found using fish edges. Approximation of middle curve gives two polynomial functions, which we combine with a help of a cubic spline approximant.

Approach is not restricted to neither fish behavior nor specific set up, and later research is needed to discover other choices of using proposed method of a surface estimation.

REFERENCES


Taverna Laura; Chellali R.R.L., 2010. 3D simulation of robotic fish interactions with physic-based underwater environment. In OCEANS 2010. IEEE.


In Robotics, Automation and Mechatronics. IEEE.


COMPLEX INTERMODAL TRANSPORT SYSTEMS
A GRAPH MODEL OF INTERMODAL TRANSPORTATION NETWORKS

Rui P. P. Cardoso, Antonio D. Casimiro, Rosaldo J. F. Rossetti
Artificial Intelligence and Computer Science Lab
Department of Informatics Engineering
Faculty of Engineering, University of Porto
Rua Dr Roberto Frias, s/n, 4200-465, Porto, Portugal
{ruippeixoto, antonio.casimiro, rossetti}@fe.up.pt

KEYWORDS
graph, network, intermodal, transportation, best path, Dijkstra, A*, Fibonacci

ABSTRACT
Nowadays, most cities have an intermodal public transportation network. It can be a challenge for one to determine an optimal path between two stations, whatever the criteria chosen. The aim of this research is to address the problem of planning such a trip, comparing two renowned algorithms.

INTRODUCTION
This paper seeks to address the problem of finding an optimal path between two points of an inter-modal transportation network. A network consists of a set of points, also called nodes, and a set of links connecting pairs of points, also called arcs. It is, therefore, often abstracted by the mathematical concept of graph[7]. Each node in the network represents a station or a stop of a public transportation line, has an associated position in the xOy-plane, and an average waiting time. Each arc represents a route between two nodes, meaning that the traveller is traversing a stretch of a given transportation line, or walking between two nodes in order to change lines. An arc has an average travel time and a travelled distance, both of which may be considered as parameters for calculating the cost of traversing the network. Derived from these, an arc also has an average speed of traversal. Note that this text deals exclusively with directed networks.

Objective function
The goal is to find an optimal path which connects node \( \nu_i \) to node \( \nu_j \). Let \( t_{\nu_i, \nu_j} \) represent the directed arc (or edge) connecting node \( \nu_i \) to node \( \nu_j \), and let \( d_{\nu_i, \nu_j} \) represent the length of the arc \( \epsilon_{\nu_i, \nu_j} \). Let \( t_{\nu_i} \) be the average wait time associated with node \( \nu_i \), and let \( t_{\nu_i, \nu_j} \) be the average time necessary to traverse the arc \( \epsilon_{\nu_i, \nu_j} \). Let \( c_{\nu_i, \nu_j} \) be the number of line changes necessary to travel between consecutive nodes \( \nu_i \) and \( \nu_j \). \( A_{i, j} \) represents the set of nodes in a certain path connecting node \( \nu_i \) to node \( \nu_j \), and \( B_{i, j} \) represents the set of arcs in a same path. Three possible definitions for the objective function are the following:

- Minimise \( \sum_{(\nu_i, \nu_j) \in B_{i, j}} d_{\nu_i, \nu_j} \), i.e., minimise the total travelled distance.
- Minimise \( \sum_{\nu_i \in A_{i, j}} t_{\nu_i} + \sum_{(\nu_i, \nu_j) \in B_{i, j}} t_{\nu_i, \nu_j} \), i.e., minimise the total time necessary to go from node \( \nu_i \) to node \( \nu_j \).
- Minimise \( \sum_{(\nu_i, \nu_j) \in B_{i, j}} c_{\nu_i, \nu_j} \), i.e., minimise the total number of changes between transportation lines required to travel from node \( \nu_i \) to node \( \nu_j \).

Reduction to a shortest path problem
If the second definition for the objective function is chosen, the graph representing the network may not be used as input to the algorithms, since \( t_{\nu_i} \) has been defined as the average wait time associated with each node \( \nu_i \). The algorithms which were tested do not accept graphs with weighted nodes as input, so the problem must be rewritten in order to make them applicable. This can be achieved by preprocessing the network in order to eliminate weighted nodes. Note that a node \( \nu_i \) with an average wait time \( t_{\nu_i} \) greater than zero can be collapsed into two nodes, \( \nu'_i \) and \( \nu''_i \), which share the same position in the xOy-plane, each with a null average wait time, and connected by an arc of length zero and travel time \( t_{\nu'_i, \nu''_i} = t_{\nu_i} \).

DIJKSTRA’S ALGORITHM
The first shortest path algorithm which was taken into account in this paper was the renowned Dijkstra’s algorithm[2, 3]. Please refer to the bibliography describing this algorithm.

Time efficiency of Dijkstra’s algorithm
Considering a priority queue for sorting the nodes in the network by the least distance from the start node, and \( |E| > |V| \), the running time of the algorithm is
\(O(|E| \cdot \log(|V|))\).

**Fibonacci Heaps**

Using a Fibonacci heap instead of a binary heap could lead to better performance, at least on theoretical grounds\([1]\). If Dijkstra’s algorithm is implemented with a Fibonacci heap, and \(|E| = O(|V|)\), the running time of the algorithm is \(O(|V| \cdot \log(|V|))\). If it verifies that \(|E| > |V|\), this is an improvement over the running time which can be achieved with binary heaps. From a theoretical point of view, Fibonacci heaps are more desirable than binary heaps under certain conditions. In practice, however, their programming complexity is usually a deterrent to their use in most applications\([1]\). It is also unclear whether they perform better in practice than binary heaps\([8]\).

**THE A* ALGORITHM**

Given the subject of this paper, one is interested only in finding the shortest path between two nodes in the network, so it seems that Dijkstra’s algorithm performs extra work. An alternative is the A* search algorithm, a goal-directed search algorithm which makes use of heuristics\([4, 3]\). If the algorithm expands nodes which obviously can’t be in the shortest path between the source and destination nodes, it is wasting resources. On the other hand, if it overlooks nodes which might be in such a path, then it is no longer _admissible_, i.e., it is no longer guaranteed to find an optimal path\([5]\). The algorithm makes use of an evaluation function, \(f^*(\nu_i)\), which estimates the cost of an optimal path from the start node to the goal node, _constrained_ to go through node \(\nu_i\).

**The evaluation function**

Let \(f(\nu_i)\) be the actual cost of an optimal path connecting a start node \(s\) to a certain goal node, and _constrained_ to go through node \(\nu_i\). An estimate of \(f(\nu_i)\) may be used as the evaluation function of the cost of such a path\([5]\). This evaluation function will be called \(f^*(\nu_i)\), and can be defined as \(f^*(\nu_i) = g^*(\nu_i) + h^*(\nu_i)\). \(g^*(\nu_i)\) can be defined as the cost of the least costly path found so far between the start node \(s\) and node \(\nu_i\)\([5]\). \(h^*(\nu_i)\), an estimate of \(h(\nu_i)\), which is the cost of an optimal path from node \(\nu_i\) to a goal node, is the heuristic function, upon which the admissibility and the optimality of the A* algorithm depend, as discussed in the next subsections.

**The admissibility of the A* algorithm**

It can be proved that, if \(h^*(\nu_i)\) is always a lower bound of \(h(\nu_i)\), then the A* algorithm is admissible, as it always finds an optimal path from the start node to a goal node. Let us define two heuristic functions, one for the application of the A* algorithm for finding the path with the least travelled distance, and another for finding the path with the least travel time. Let \(d^*(\nu_i)\) be defined as the Euclidean distance between node \(\nu_i\) and a goal node of the start node \(s\), and let \(d(\nu_i)\) be the actual distance between those two nodes on an optimal path. It is true that \(d^*(\nu_i) \leq d(\nu_i), \forall \nu_i \in V\), by the definition of Euclidean distance. Therefore, the condition is satisfied, and A* is admissible in this case. Let \(t^*(\nu_i)\) be an estimate of the travel time between node \(\nu_i\) and a goal node of \(s\) on an optimal path, which will be called \(t(\nu_i)\). If \(t^*(\nu_i)\) is based on the Euclidean distance between those two nodes, \(d^*(\nu_i)\), and on the _average speed of the fastest road_ in the network, denoted by \(\bar{v}_\text{max}\), then \(t^*(\nu_i) \leq t(\nu_i), \forall \nu_i \in V\). The condition is verified, and A* is also admissible in this case.

**The optimality of the A* algorithm**

If \(h^*(\nu_i)\) is any lower bound on \(h(\nu_i)\), then A* is admissible. A possible lower bound could be \(h^*(\nu_i) = 0, \forall \nu_i \in V\). If a higher lower bound of \(h(\nu_i)\) is used for \(h^*(\nu_i)\), then A* is still admissible, but fewer nodes will be expanded\([5]\). Consider the consistency assumption \(h(\nu_i, \nu_j) + h^*(\nu_j) \geq h^*(\nu_i)\). This assumption, when verified, means that an estimate \(h^*(\nu_i)\) would not be improved by the situations represented by other nodes\([5]\).

Both heuristics, \(d^*(\nu_i)\) and \(t^*(\nu_i)\), verify the consistency assumption. If a heuristic satisfies the consistency assumption, then the A* algorithm does not do repeated work, and is optimal, in the sense that no other admissible algorithms expand fewer nodes, as long as they use no more information from the problem domain than A* does\([5]\).

**Time complexity of the A* algorithm**

The complexity of A* depends on the choice of the heuristic function, as not only might its calculation contribute to the global complexity of the algorithm, but it may also influence how the algorithm behaves. If the algorithm makes use of a binary heap, and \(|E| = O(|V|)\) and \(|E| > |V|\), its time complexity is \(O(|V| \cdot \log(|V|))\). If, instead, a Fibonacci heap is used, and \(|E| = O(|V|)\), then the time complexity of the algorithm is \(O(|V| \cdot \log(|V|))\).

The time bounds of the A* algorithm, using either one of the heuristics \(d^*(\nu_i)\) and \(t^*(\nu_i)\), and Dijkstra’s algorithm are the same. It was claimed that Dijkstra’s algorithm might perform extra work when searching for an optimal path between a pair of nodes. However, there is no theoretical evidence that A* performs better.

**COMPARING PERFORMANCES**

Figure 1 and Figure 2 show how the running time of the two algorithms, using both a binary heap, and a
Fibonacci heap, grows as $|V|$ and $|E|$ grow, respectively.

![Running Time of 1000 Repetitions of the Algorithms](image)

Figure 1: Time complexity of algorithms as $|V|$ grows.

Each vertical group of dots in the sequence represents the same network in both charts. Although they appear quite efficient from a theoretical perspective, it looks like Fibonacci heaps perform worse than binary heaps, in practice. The reasons for this have been discussed in previous sections.

![Running Time of 1000 Repetitions of Algorithms](image)

Figure 2: Time complexity of algorithms as $|E|$ grow.

As far as the algorithms are concerned, Dijkstra’s performed slightly better than the $A^*$ algorithm as $|V|$ and $|E|$ grow. However, the running times are very close, and the results are, thus, inconclusive. Having stated that the theoretical time bounds for both algorithms are the same, no empirical evidence has been found that either $A^*$, or Dijkstra’s algorithm performs better. Note also that the performance of the algorithms is related to how many nodes are expanded before finding the goal node, which depends greatly on the problem domain. It is therefore the authors’ judgement that the choice of which algorithm to resort to must be based on careful analysis of the problem, resulting in as most informed a decision as possible.

**CONCLUSIONS AND FUTURE WORK**

The problem of planning a trip in an inter-modal transportation network has been reduced to a shortest path one. The motivation behind the research on the $A^*$ algorithm was the fact that Dijkstra’s algorithm might have been performing extra work when calculating an optimal path between two nodes in the network. In that sense, $A^*$ was tested to see whether it could speed up the process by using heuristics capable of leading straight towards the goal node. However, no evidence has been found of that. Both of these algorithms make use of priority queues, implemented with a binary heap or with a Fibonacci heap. Two versions of each algorithm were tested, one for each data structure, which lead to conclude that binary heaps were able to outperform Fibonacci heaps. The authors hope that this paper will lead to further research on multiple subjects, such as the problem of route assignment, development of games related to transportation networks [6] and other fundamental issues which smart cities’ mobility agenda nowadays must address.

**References**


VERIFICATION BY DISCRETE SIMULATION OF INTERLOCKING SYSTEMS

Quentin Cappart
Christophe Limbrère
Pierre Schaus
Université catholique de Louvain
Place de l’Université 1, 1348, Belgium
E-mail: quentin.cappart@uclouvain.be

Axel Legay
INRIA/IRISA, Rennes, France

KEYWORDS
Discrete event simulation, Verification, Railway, Interlocking

ABSTRACT

In the railway domain, an interlocking is the system controlling active components in a station in order to ensure a safe train traffic. The behaviour of modern interlockings is defined by particular data, called application data, describing the actions that the interlocking can take and under which conditions. However, application data are either prepared manually or prepared automatically by tools that do not guarantee a sufficient level of safety. Given the high level of safety required by such a system, the verification of the application data is a critical concern. Recent researches dealing with this issue are based on model checking. Due to the state space explosion problem, this approach does unfortunately not scale for large stations. In this paper, we present an innovative approach for the verification of interlocking data, based on a discrete event simulation, which does not suffer of the state space explosion problem. Although sacrificing exhaustiveness, we show experimentally on a real life instances that this approach is able to detect any introduced errors in the application data within seconds.

INTRODUCTION

Each train station is controlled by a system having the responsibility to ensure safe movements of trains and to avoid all risk of conflicts between their path. Such a system is called an interlocking. More specifically, an interlocking controls the physical components of the infrastructure such as the points and the authorities of movement (e.g. signals) in order to safely allow the trains through a station.

Unlike the interlockings of the first generation that were based on a mechanical or relay logic, modern interlockings are computer based which means that the actions are calculated by a software. Furthermore, the software of computer-based interlockings relies on configuration data specifying what are the possible actions and under which conditions they can be taken (Theeg et al., 2009). Such data are called application data and are specific to each station.

The safety of the train traffic inside a station is thereby highly dependent on the correctness of its application data. However, the application data are prepared manually and are thus subject to human errors leading their verification to a critical concern. One approach for interlocking verification deeply studied in the literature is model checking (Huber and King, 2002; Winter et al., 2006; Busard et al., 2015). The goal of a model checker is to verify if a system meets a set of safety properties by considering all the reachable states of the model representing the system. This method is exhaustive, in other words, if a requirement is not satisfied, it will always be detected.

However, model checking suffers from the state explosion problem (Clarke et al., 2012). Whereas small sized stations can be verified efficiently the verification time grows exponentially as the size of the station increases and might not return a result within a reasonable time in practice. This is a well known limitation for model checking. Several techniques to limit this problem have been proposed. Winter and al. (Winter et al., 2006) propose to relax the verification by reducing the complexity of the model and to improve the verification process by using the properties of the system. In (Huber and King, 2002), Huber and King implemented a symbolic model checker with different optimisations like a dynamic variable re-ordering. Winter (Winter, 2012) also proposes several strategies to optimise the variable ordering. Eisner (Eisner, 1999) uses symbolic model checking. Busard and al. (Busard et al., 2015) proposed in their model a customized model-checking algorithms based on operation on the BDD.

Despite these optimisations, applying model checking on medium or large stations remains intractable. In (Busard et al., 2015) only very small stations could be verified within hours of computations. To overcome this issue, we propose in this paper a new approach
for the verification based on discrete event simulation (Schriber et al., 2012) which does not suffer from the state space explosion problem but sacrificing the exhaustiveness property. We show experimentally on a medium size real life instances of a Belgianinterlocking that this approach is able to detect any introduced errors in the application data within seconds.

Usage of simulation in the railway domain is not new. The company OpenTrack provides a railway simulation tool (Nash and Huerlimann, 2004) to verify the capacity of the railway network, the feasibility of the schedules, collect statistics about running times, etc. but which is not related to interlocking verification. Hon and Kollmann (Hon and Kollmann, 2006) proposes an hybrid model for the verification based on simulation coupled with model checking. For the simulation, they use the software Rhapsody (Gery et al., 2002) which takes test cases as input and check if a system is correct by simulating the test cases. However, the test cases must be elaborated manually and it turns thereby into a manual verifications which is different of what we want to do.

To the best of our knowledge there is no existing work considering a discrete event simulation for verifying an interlocking system. Our methodology can be used on any application data formats used in any country.

In this work we instantiate our approach to the SSI format (Cribbens, 1987) mainly used by Infrabel (www.infrabel.be) in Belgium. In the next section, we describe how an interlocking works on a real life interlocking instance. This case study corresponds to a medium sized railway interlocking system of a Belgian station. In Section 3, we explain the principles and the benefits of our discrete event simulation for the verification. Finally, in Section 5, we discuss the experimental results obtained on the case study.

INTERLOCKING PRINCIPLES

As previously said, the role of an interlocking is to prevent any conflicting movements while the trains move on their reserved routes in the station. This section explains how it is done in practice. To do so, let us first consider a case study, the station of Braine l’Alleud, which is a typical Belgian medium sized station. Figure 1 shows the track layout of Braine l’Alleud (all the variable names are not represented on the Figure).

On this figure, several elements can be identified:

- The **points** (e.g. P.01AC) are the movable components that allow trains to move from one track to another one. According to the Belgian convention, a point can be in a normal position (left) or in a reverse position (right).
- The **track segments** are portion of the track where a train can be detected. They are either occupied, or clear. Track segments are delimited each other by the **joints**.
- The **physical components** are controlled and monitored by the interlocking. Besides, the interlocking software makes use of the **logical components**:
  - The **routes** are the paths that the trains follow when running through the station. Each route starts from a signal and finishes to another signal or to a track segment. In the application data, the routes have the following format: R_src.dest (e.g. R_CC_104) with src the name of the start signal and dest the name of the destination. A route can either be set if it is reserved for a train, or unset on the contrary.
  - The **subroutes** are the contiguous segments that the trains follow inside a route. When a route is commanded for a train, a set of contiguous subroutes is locked establishing a path from the origin of the route to its destination. When not requested, subroutes are in a free state. In the application data, they have the following syntax: U_src.dest (e.g. U_KXC_20C).
  - The immobilisation zones, also called **UIR**, are the components materialising the immobilisation (locking), of a set of points for a route. As for the subroutes, they can be on a locked or a free state depending on whether they are reserved for a route. If they are locked, the points attached to the UIR are not supposed to move. In the application data, they are presented like this: U_IR(identifier) with identifier the name of the UIR. Generally, the identifier is related to the name of the points locked by it. For instance, U_IR(08BC) locks Point P.08BC.

Using these components, the interlocking can control the train traffic by commanding the routes. The actions that have to be done and the conditions under which they can be executed are described in the application data. To explain the content of the application data, let us consider the scenario where the route from the signal KC to the track 103 has to be set:

1. Firstly, the interlocking will verify whether the request for the route R_KC_103 can be granted.

\[
\begin{align*}
1 & \text{ //Q_R(KC_103)} \\
2 & \text{ if } R_{KC_103} \text{ xx} \\
3 & \text{ P_08C cfr, P_08AC cfr,}
\end{align*}
\]
Figure 1: Layout of Braine l’Alleud Station

Listing 1: Request for Setting Route R_KC.103

Listing 1 presents a typical route request as described in the SSI format. There is a similar request description for each route that can be commanded in the station. The first part of this request (line 2 to 6) are the conditions under which the request can be granted. More specifically, it can be granted if the route is not already set (xs value on line 2), if some points are free to be commanded to the reverse (cfr) or to the normal (cfn) position (line 3 and 4), and if some UIR are in a free state (f on line 5 and 6). The components requested can be seen on Figure 1.

2. Before moving a point, the interlocking must ensure that it can be moved without causing safety issues. Such conditions are also expressed in the application data.

1 #P_08BCN U_IR(08BC) f // normal position
2 *P_08BCR U_IR(08BC) f // reverse position

Listing 2: Conditions Allowing Point P.08BC to Move

Listing 2 states that the point P.08BC can be controlled in the normal position only if its UIR is free. The same condition is verified to control it in the reverse position. These conditions are the route setting conditions. If they are not satisfied, the route request is dropped.

3. If each condition is satisfied, the route can be set and the actions defined in line 7 to 13 of Listing 1 are taken. More precisely, the route is thoroughly set (s on line 7), the points are commanded either to reverse (cr) or to normal (cn) position (line 8 and 9), the UIR (line 10 and 11) and the subroutes (line 12 and 13) are locked. The home signal of the route (KC in our example) is controlled at its proceeding state when additional conditions are fulfilled. These additional conditions are abstracted in our model. At this step, the train can run through the station.

4. While the train is running through the route, reserved component can be progressively released. For route R_KC.103, when the train has cleared Track containing P.08BC, UIR(08BC) can be released. The conditions under which components can be released are also described in the application data.

1 U_KC.19C f
2 if R_KC.102 xs, R_KC.103 xs,
3 R_KC.104 xs,
4 T.08BC c

Listing 3: Subroute U_KC.19C release conditions
Listing 3 states that Subroute U.KC.19C can be released whether the routes on (line 2 and 3) are not set and the Track segment T.08BC (line 4) is free.

1 if U_IR(08BC) then
2   if U_17C_KC f,
3       U_KC_17C f,  
4       U_19C_KC f,  
5       then U_IR(08BC) f

Listing 4: Conditions for Releasing UIR(08BC)

Listing 4 shows the release conditions for UIR(08BC): if it is locked (line 1), then it can be released (line 6) only if some subroutes are not locked (line 2 to 5). Unlike the reservation actions which are only executed upon request, the releasing actions areperiodically verified.

This process briefly describes the route cycle controlled by the interlocking. To be more precise, real interlockings contains other components and other actions which are abstracted in our study. As previously said, verification of application data is a crucial task: an error or an omission can lead to serious safety issues. For instance, let us assume that the action P.08BC cr on Listing 1 is transformed on P.08BC cn. Following Figure 1, the train will move through Track T.07AC instead of Track T.08AC which can lead to a head to head collision from a train following Route R.JXC.012. There is thereby a real need of efficient and reliable methods to verify the application data.

VERIFICATION BY SIMULATION

On this section, we present a novel approach based on a discrete event simulation which does not suffer of the drawbacks of model checking. The idea is to simulate the train movements and the behaviour of an interlocking as described in its application data and to observe if any safety issues occurred. If no issue occurred and if the simulation time was long enough, we can have a high expectation that the system is safe. Compared to model checking where all the states are considered even the ones corresponding to cases that never occur in practice, the discrete simulation will only consider the cases which can potentially happen with a real interlocking.

A discrete event simulation involves three kinds of components:

- The entities, which are the active objects on which the simulation is applied. Each entity is characterized by its current state. Our model contains two families of entities: the interlocking components described in the previous section (e.g. the subroutes which can be free or locked), and the trains, characterized by their position.

- The events, which define actions that can alter the state of the entities and which can generate other events. On one hand, there are events for the actions defined in the application data: requesting a route and releasing a component. The execution of these actions is guarded by their conditions. The effect of these events is to set the considered entities into the requested state. On the other hand, there are the events triggered by the train movements: entering into the station, moving through it and leaving it.

- The clock, stating when the events must occur. Unlike a continuous simulation where event can occur during a time period, the discrete simulation requires each event to occur at a particular instant.

In our simulation, all of these components interact together as follows:

1. Trains randomly arrive in the station at the possible routes home signals. In practice, a train arrival is an event which can occur with a uniform probability on the discrete time interval \([t_a, t_a + n_a]\) where \(t_a\) is the time of the last train arrival (\(t_a = 0\) for the first step) and \(n_a\) is a predefined parameter. Besides, each time such an event occurs, a new event is triggered in the interval \([t_a, t_a + n_a]\) while \(t_a\) is updated.

2. Route requests are periodically issued for the trains waiting at a start signal. If the route setting conditions are fulfilled, the route is set and all the actions described in the request are executed. Otherwise, the request is discarded and no action is taken. Like the trains arrival, a route request is an event which can occur in an interval \([t_r, t_r + n_r]\) with \(t_r\) the time of the last request and \(n_r\) a predefined parameter.

3. Trains move through the station following the path described by the station components state. Concretely, they always move forward from one track segment to the next one and follow the direction defined by the position of the points. The first movement of a Train \(x\) is triggered when its attached route request is accepted. The next movements occur in the interval \([t_m(x), t_m(x) + n_m]\) with \(t_m(x)\) the time of the last movement done by Train \(x\) and \(n_m\) a parameter. Each train has thereby its own queue of events. By doing this, we implicitly model the fact that the speed of the trains can be different. The higher is \(n_m\), the larger will be the speed difference between trains. Modelling an exact or a realistic speed has no importance for the verification. What matters is to have a \(n_m\) large
enough to allow the simulation to cover all the possible combinations of train positions. To do so, \( n_m \) must be higher than the largest number of movement steps for a train (i.e. the number of track segments composing the longest route). Moreover, the train lengths are abstracted, only the occupation of the track segments has an importance for the verification.

4. After each train movement, the system checks if a releasing event can be triggered. In this case, the requested components are thoroughly released.

5. When a train reaches the end of a route, it is removed from the station.

To model the randomness in our simulation we introduced parameters \( n \) for several kinds of events. Their goal is to define the time steps range on which the events can occur. Therefore, the values of \( n \) determines the frequency of occurrence of a family of events: the lower is \( n \), the higher will be the frequency. Besides, the exact values of \( n \) has no importance, what matters is the relation between the values. A \( n \) lower than others indicates that the events related to \( n \) will have a higher probability to occur than the others. We choose to assign the same value than \( n_m \) for each \( n \), which means that each event has the same probability to occur. As we will see in the next section, this default value provides good results. Figure 2 shows two possible scenarios for the event sampling for two trains with \( n = 5 \).

The model presented here is implemented in Scala using the discrete event simulation package of OscaR (OscaR Team, 2012). This toolkit has similar functionalities as SimPy (Müller and Vignaux, 2003). Once the simulation is launched, we can observe the expected behavior of the interlocking system as described by its application data and how it allows the trains to move through the station. The analysis of this behavior is finally used to verify the correctness of the application data. The key idea to perform the verification is to monitor the simulation to see whether a situation causing a safety issue has occurred. The monitoring can be done in two ways: with a dynamic GUI displaying the routes and the train movements in real time, or with the execution trace summarizing all the actions performed during the simulation. As identified in (Busard et al., 2015) there are three conflictual situations:

- A train moves through a point not set in a position, allowing the train to continue its path. For instance, in Figure 1, it occurs if a train follows Route R.CC.102 and if Point P.02BC is not set at the reverse position. It will also cause a derailment.

These situations can be expressed in terms of the state of the entities, and are easy to detect. Using this simulation, we can thereby verify in a non-exhaustive way the correctness of the application data.

VALIDATION AND PERFORMANCE

In this section, we analyse the results obtained in order to validate the approach and its performance in terms of speed, test coverage and error detection capability.

The results presented here are based on two instances. A small sized station, Namèche, containing 7 points, 7 UIR, 7 signals, 14 routes and 26 subroutes. The other instance is our case study, Braine l’Alleud (Figure 1), which is a medium sized station containing 12 points, 10 UIR, 12 signals, 32 routes and 48 subroutes. Furthermore, we compared the results with the ones obtained using Busard et al. method (Busard et al., 2015) which made experiments on Namèche with a model checking approach.

Table 1: Benchmark about Execution Time (in seconds) of Error Detection in seconds

<table>
<thead>
<tr>
<th></th>
<th>Namèche</th>
<th>Braine l’Alleud</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simulation</td>
<td>2.78 / 0.10</td>
<td>8.73 / 2.23</td>
</tr>
<tr>
<td>Model checking</td>
<td>370</td>
<td>242</td>
</tr>
<tr>
<td>1.</td>
<td>2.94 / 0.45</td>
<td>6.88 / 2.57</td>
</tr>
<tr>
<td>2.</td>
<td>3.09 / 0.32</td>
<td>7.05 / 0.09</td>
</tr>
<tr>
<td>3.</td>
<td>3.32 / 0.19</td>
<td>6.71 / 1.28</td>
</tr>
<tr>
<td>4.</td>
<td>2.98 / 0.06</td>
<td>6.49 / 0.88</td>
</tr>
<tr>
<td>5.</td>
<td>169</td>
<td>1.85</td>
</tr>
</tbody>
</table>

Table 1 recap the execution time in seconds for detecting the following errors with the simulation and the model checking approach:

1. Missing condition in a route request.
2. Point moved to a wrong position when setting a route.
3. Subroute not properly locked when setting a route.
4. Condition missing for releasing a subroute.
5. Condition missing for releasing a UIR.
Given the randomness of the simulation approach, we repeated each experiment five times by introducing five different errors of each kind. The time \((x/y)\) presented in Table 1 corresponds to the arithmetic mean \((x)\) and the standard deviation \((y)\) between their execution time.

We can make several observations from this table. Firstly, we can see that the simulation approach detects thoroughly errors in the application data, even if exhaustiveness is not guaranteed. However, it is important to mention that the application data are robust by their design. Several forms of redundancy are implicitly comprised inside them which implies that an inconsistency do not irremediably causes a safety issue. For instance, removing an UIR statement in the route setting conditions can be covered by another contiguous condition. Therefore, introducing errors causing safety issues becomes an harder task than expected.

Besides, we can also observe that our approach detects errors significantly faster than the model checking approach. Concerning the scalability of the approach, the benchmark for Braine l’Alleud shows that the execution time for a larger station do not increase that much compared to Namèche. Although experiments on larger data set have not been done yet, the possibility of a full parallelisation strengthens the scalability of the approach.

Indeed, the verification by simulation can be almost entirely paralleled without any overhead if the simulation time is not too short. The intuition behind this assumption is that the train traffic occurring today does not influence the train traffic occurring 10 days later. In other word, a simulation covering 20 days is identical to two simulation of 10 days. The only overhead occurring is due to the time used to parse the application data and to generate the simulation model which can be neglected. By Gustafson’s law (Gustafson, 1988), we can thereby deduce that with \(n\) processors, the execution time should be divided by \(n\).

Furthermore, before using simulation for the verification, we need to decide during how much time the simulation must be active in order to have a high expectation that the potentially errors will be successfully discovered. The first step is to know what is the correspondence between the simulation time and the real time or in other words, how many hours are covered for a simulation of one hour. To do so, let us assume the worst case of a busy station where there is an incoming train every minute during a whole day (1440 trains per day). By recording the number of routes set during the simulation, we can report the number of trains that have moved through the station and deduce how many days the simulation has covered. Following this procedure, we obtain that 1 hour of simulation for Braine l’Alleud covers approximately 16 872 days (\(\approx 46\) years) of real interlocking operations. A simulation of 1000 years will thereby takes 22 hours of computation without resorting to parallelisation. However, the results of Table 1 shows that in practice error detection is done far more quickly that the time required to reach this period.

The major drawback of our approach is that there is no guarantee of exhaustiveness. Therefore, it is theoretically possible that there exist conflictual scenarios not covered by the simulation. To gain confidence about our model, we have designed covering tests
aiming to measure which scenarios are tested. For an interlocking system, a scenario corresponds to a route request accepted whereas the station has a particular configuration. For instance, the Request Q(R.CC.102) of Listing 1 can be made when none or several routes are already set in the station. Furthermore, a same route can have different states depending on which of its elements are released. Similarly to software testing where code coverage (Ammann and Offutt, 2008) is used to gain confidence into the quality of test suites, we also measure and report statistics related to the scenarios coverage. More exactly we record for each request the number of times it is generated, granted, and which routes were already set when granted.

The idea behind this test coverage is twofold. First, it aims to verify that the requests can be done in many different situations and secondly, it can be used to detect conflictive routes. Table 2 summarizes this test coverage for the scenarios where a request is done when Route R.CXC.104 is set. After 1 hour of simulation, 361,496 requests were done under this assumption. Furthermore, each scenario occurred with a uniformly probability with a mean of 11661 and a standard deviation of 180.

<table>
<thead>
<tr>
<th>Request</th>
<th>#Granted</th>
<th>#Done</th>
<th>Request</th>
<th>#Granted</th>
<th>#Done</th>
</tr>
</thead>
<tbody>
<tr>
<td>DXC.092</td>
<td>0</td>
<td></td>
<td>EC.091</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>KXC.103</td>
<td>22.65</td>
<td></td>
<td>CC.102</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>CC.103</td>
<td>0</td>
<td></td>
<td>CGC.012</td>
<td>13.70</td>
<td></td>
</tr>
<tr>
<td>KC.101</td>
<td>24.37</td>
<td></td>
<td>CC.104</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>KC.101</td>
<td>24.85</td>
<td></td>
<td>KXC.101</td>
<td>24.13</td>
<td></td>
</tr>
<tr>
<td>KC.102</td>
<td>24.36</td>
<td></td>
<td>IC.011</td>
<td>14.51</td>
<td></td>
</tr>
<tr>
<td>EC.092</td>
<td>0</td>
<td></td>
<td>FC.091</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>KXC.104</td>
<td>21.89</td>
<td></td>
<td>KC.104</td>
<td>23.11</td>
<td></td>
</tr>
<tr>
<td>DC.091</td>
<td>12.62</td>
<td></td>
<td>KC.103</td>
<td>22.78</td>
<td></td>
</tr>
<tr>
<td>JXC.011</td>
<td>16.16</td>
<td></td>
<td>CXC.101</td>
<td>12.71</td>
<td></td>
</tr>
<tr>
<td>CXC.102</td>
<td>0</td>
<td></td>
<td>JXC.012</td>
<td>14.75</td>
<td></td>
</tr>
<tr>
<td>JXC.012</td>
<td>16.33</td>
<td></td>
<td>DXC.091</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>CGC.011</td>
<td>13.69</td>
<td></td>
<td>CXC.103</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>KXC.102</td>
<td>24.15</td>
<td></td>
<td>DC.092</td>
<td>12.57</td>
<td></td>
</tr>
<tr>
<td>IC.012</td>
<td>14.05</td>
<td></td>
<td>FC.092</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>JC.011</td>
<td>15.38</td>
<td></td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

This table summarises the proportion of times (in percent) that a request is granted after being issued. We can observe that some requests, like Q(R.DXC.092), are always refused when Route R.CXC.104 is set. In Figure 1 we can indeed notice that R.DXC.092 is highly interleaved with R.CXC.104 such that there exists no state of R.CXC.104 where R.DXC.092 can be also set.

For the other requests, they are all much less often granted than they are done. It is because other routes can also be set in the station, which will prevent the acceptance of the request. However, we can notice that some routes have a lower probability to be set than other. It corresponds mainly to the routes beginning in the middle of the station (from Signals DC, DXC, ED, IC, CGC, JC or JXC in Figure 1). It is because on such locations, trains can have a route going either to left, or to right. Therefore, the probability to have a route going to a particular direction is reduced. Furthermore the requests having the lowest probability to be granted are Q(F.CXC.101), Q(R.DC.091) and Q(R.DC.092) which are all three interleaved with R.CXC.104. Generally speaking, the more a route is constrained, the lower will be its probability to be set. Similar results are observed for scenarios involving other routes, which shows that most of the scenarios are covered by the simulation.

Through the analysis of the results presented in this section, we thereby confirmed the validity of the simulation approach, its performance, and its benefits.

**CONCLUSION**

Verification of an interlocking system is a safety concern. Up to now, most of the research work considered for the verification were based on model checking. However, due to the state space explosion, this approach does not scale very well for large stations. In this paper, we presented a new approach of verification based on a discrete event simulation aiming to address the problems encountered by the model checking. We have first described the principles of this approach and how it can be used for a verification purpose. Besides, we have confirmed through experimental results the validity of this approach and shown that it does not suffer from the problems of the model checking approach. Furthermore, we presented other benefits of the method as the possibility of a full parallelisation in order to do a long-term simulation.

The present work focuses on defining the discrete event simulation. However, it does not show how the results of performing several simulations can be aggregated to obtain a confidence on the reliability of the entire system.

As a future work, we plan to plug our approach in a statistical model checker (SMC). The idea behind SMC is to perform several simulations of a given system, and then to use statistics (Monte Carlo, hypothesis testing, etc.) in order to reveal information on its global behavior (Legay et al., 2010; Younes and Simmons, 2006). Classical SMC approaches are well-suited to compute the probability that a system satisfies a given property. Here, we are more interested in guiding the simulation so that it reveals a rare failure of the system. Our knowledge of the system will be used to force the simulation to
reach a failure, and to stop it if there is enough evidence that it will not reach a failure.

ACKNOWLEDGEMENT

This research is financed by the Walloon Region as part of the Logistics in Wallonia competitiveness pole.

REFERENCES


PROVIDING ADAPTIVE TRAFFIC ROUTING BASED ON USER AND NETWORK CONTEXT

Sergio S. Spinola, André L. C. Oliveira, Joatham P. S. Silva, Carlos F. J. Muakad, Paulo N. M. Sampaio

Computing and Systems Graduate Program
Salvador University (UNIFACS)
Salvador, BA - Brazil
email: srgspnl@gmail.com, andreliuzoliveira@gmail.com, joathamp@gmail.com, fred.fredjansen@gmail.com,
pnms.funchal@gmail.com.

Lina M. P.L. Brito
Madeira Interactive Technologies Institute (M-ITI)
University of Madeira
Funchal 9020-105, Portugal
email: lina@uma.pt

KEYWORDS
Adaptive routing, quality of service, quality of experience, context, quality of context, context-based routing.

ABSTRACT

Providing real-time traffic guarantees and fairness based on the availability of network resources has been a major issue presented in the literature. However, due to the convergent nature of digital architectures, the increasing demand of upcoming real-time sensitive traffic, such as VoIP, and a higher user’s adaptability (devices, global positioning, content quality, etc.), solutions based on Quality of Service (QoS) turned out to be insufficient in order to meet user’s requirements. Indeed, QoS metrics are network-centered, and mostly related to the dynamic nature of the traffic (such as throughput, delay, jitter, among others). In order to meet the need for a user-centered network, this paper proposes a context-aware solution where the concepts of Quality of Service, Quality of Experience and Adaptive Routing are integrated in order to provide a more dynamic and pro-active approach for the delivery of context-oriented time-sensitive traffic.

INTRODUCTION

The intensive use of the current IP networks requires an optimized management of the available resources of the network infrastructure in order to enable the co-existence of multiple types of traffic. On the last few years there has been a considerable increase in bandwidth availability, which motivated a burst regarding the proposal of new applications, presentation devices, mobile communication, etc. In particular, we should consider emerging applications and services (such as Voice over IP – VoIP – and videoconferencing), which generate an increasing amount of real-time data traffic to the network. Unfortunately, network infrastructure and routing strategies have not evolved at the same pace as data applications. Therefore, network infrastructure is constantly under resources shortage and consequently under congestion.

Different contributions have been proposed in the literature in order to provide real-time traffic guarantees and fairness concerning the utilization of network resources.

Therefore, Quality of Service (QoS) has been the key solution in order to meet user’s requirements. For this purpose, most of the current QoS contributions rely on the concepts introduced by classical paradigms such as Integrated Services (IntServ), Differentiated Services (Diffserv), MPLS/GMPLS and Traffic Engineering (Balci and Sargent 1981).

Currently service providers have deployed these QoS techniques in order to determine configuration strategies, planning and provisioning network services. These techniques are related, in general, to admission and congestion control, buffer management and scheduling. However, regardless of the techniques applied the expected delivery quality has not been achieved completely, generating consumer dissatisfaction with the services offered (El-Genidy et al. 2003). Although Service Level Agreement (SLA) establishes users and infrastructure parameters for the delivery of a particular traffic, the dynamic nature of the user and application’s environment should also be considered.

In order to embody the concept of a user-centered network, the notion of context can be applied. Context awareness is understood as an ubiquitous and/or pervasive computing paradigm that aims at dealing with changes in the computational system environment (Shaikh and Collange 2010). The implementation of context aware networks can be helpful in order to improve user’s experience and satisfaction when accessing network resources. For this purpose, this work addresses the proposal of a context-aware solution, which relies on three main concepts: Quality of Service, Quality of Experience and Adaptive Routing.

RELATED WORKS

Different issues should be considered when addressing context-based routing. Context is related to both user and
communication platform. Therefore, the context-based solution proposed in this paper relies on the definition of Quality of Service (QoS) (Yerima 2011), adaptive routing (Karthisia and Balamarugan 2013) and implementation of Quality of Experience (QoE) (Alreshoodi and Woods 2013). As expected, the contributions related to QoS frameworks focus on the proposal of protocols and mechanisms in order to optimize the resources availability related to network equipments. QoE, instead, refers to user’s requirements and expectations and how they actually perceived the service delivered. In order to meet user expectations, the implementation of QoS should also be centered on the perspectives of the end users. In general, QoE can be correlated by the measurement of MOS (Mean Opinion Score), whose values range among bad experience, poor, acceptable, good and excellent (El-Gendy et al. 2003).

QoE, instead, refers to user’s requirements and expectations and how they actually perceived the service delivered. In order to meet user expectations, the implementation of QoS should also be centered on the perspectives of the end users. In general, QoE can be correlated by the measurement of MOS (Mean Opinion Score), whose values range among bad experience, poor, acceptable, good and excellent (El-Gendy et al. 2003).

CONTEXT AWARE SYSTEMS

Context can be defined as "any information that can be used to characterize the situation of entities (i.e. whether a person, place or object) that are considered relevant to the interaction between a user and an application, including the user and the application themselves. Context is typically related the location, identity and state of people, groups and computational and physical objects" (Dey et al. 2001). The use of context can be applied to entities, persons, places, or even to an object relevant to the application, by the definition of characteristics of individuality, activity that may be involved, location and time and even relationships with other entities (Zimmermann et al. 2007). In order to assess context, and device, the notion of Quality of Context and Quality of Device are also considered.

Quality of Context

Although the contribution of context-aware systems can be expressive, their effectiveness can only be achieved if context information is properly defined and validated. Therefore, the definition of Quality of Context (QoC) is required in order to provide the understanding between QoC, QoS and Quality of the Device (QoD). The latter is related to the hardware components in charge to collect and provide the context information (Nazario et al. 2012), (Bucholz and Sheeffers 2003).

In order to provide QoC some metrics should also be defined, such as (Weiser, 1999): Accuracy of information (Precision); Likelihood the information is correct (Probability of Correctness); Level of trust in sources of information (Trust-worthiness); Resolution of the levels of granularity of information (Resolution); Timeliness of information related to their temporal characteristics (Up-to-dateness).

Quality of Device

Besides Quality of Context (QoC) concerning the characterization of the collected context information, it is also important to consider the Quality of Device (QoD), which is related to the precision of the computing device that will collect the context information. For instance, the Global Positioning System (GPS) of each device can have different levels of precision, or even a particular device that is not able to provide some parameters compared to another due hardware incompatibilities or the lack of ability to collect such information (Mascolo and Museli 2006). Therefore, QoD will provide information on the technical characteristics of each device and its capabilities (Vieira et al. 2009).

Most of these contributions are related to the utilization of context applied mainly to wireless networks. In these studies, the use of context allowed improvements mainly in: stability of the communication link, increased bandwidth (by decreasing overhead), higher batteries autonomy, shorter delay and scalability. These environments differ greatly from wired networks, mainly due to storage capacity and processing constraints, battery life limitation, and in some cases, limited bandwidth.

It is also important the adoption of clear policies in order to provide the correct analysis of contextual information and to be in conformance with QoC (Mascolo and Museli 2006). For this purpose, some contributions in the literature proposed solutions for improving the adoption of QoC policies based not only on the current context, but also on the effects of erroneous context information with low quality and its effects on systems, such as Proteus (Manzoor et al 2009).

PROPOSED CONTEXT AWARE SOLUTION

The conceptual definition of context information and context-aware routing enabled the proposal of a generic context-aware data routing mechanism. Thus, network devices such as routers and switches are able to choose accordingly the routing paths and traffic prioritization based on context information. This section introduces the proposed context model and architecture.

Context Aware Model

The context model applied in the proposed context-aware solution for adaptive routing can be used for both wireless and wired networks. This model is generic allowing the description of different network scenarios and the adaptation based on user’s experience.

The adopted context model describes the state of a particular entity (for instance, a user, router, switch, etc). Therefore, the following features describe this entity (Figure 1):

1. Individuality – which describes a particular information about an entity, such as identification, addressing, protocols, etc.;
2. Time – which describes time information, such as timestamp related to the status of an entity in a given instant;
3. Location – which is related to real or virtual location of an entity, and may be generated by a system such as GPS location, or by referencing information such as home, building, city, a network address, etc.;

4. Activity – which allows the description of explicit goals, tasks and actions performed by an entity, and;

5. Relations – which describes the entity's relationships with other entities, dependencies between entities, connections with objects, people, places, services, etc. Some other aspects can still enhance the description of an entity, such as:

6. Quality of Experience (QoE) – which describes a group of parameters regarding user’s perspective, which is most of times rated as MOS;

7. Quality of Device (QoD) – which describes a group of parameters regarding devices characteristics such as capabilities, computational power, precision level of data collecters, and;

8. Quality of Service (QoS) – which is related to all the metrics (qualitative/quantitative) considered on an SLA between user and platform, such as (bandwidth, delay, jitter, etc.);

The context model can also be validated according to some metrics, which determine the Quality of Context:

9. Precision – level of information accuracy to assess its relevance;

10. Probability of Correctness – assessment of the probability of the information being correct;

11. Trustworthness – assessment of the level of trust on the information source;

12. Resolution – level of granularity of a given information;

13. Up-To-Dateness – assessing how the information provided is updated

Previous works introduced XML as a representation of data as well as a Relational Database Management System RDBMS implementation for the Context Controller. In this work, in order to represent the context model, MongoDB (MongoDB Architecture, 2015) has been adopted. MongoDB is a document-oriented NoSQL database based on collections adopting a data format able to optimize end-to-end data transfer and management without the need to carry out time-consuming parsing. Therefore, a Javascript Object Notation (JSON and BSON, 2015) description turned out to be useful since it can be handled directly using programming languages, such as Python.

The modular proposal for the context model allows components to be developed independently, even though they apply and manage the same common representation for the context information, as defined in JSON/BSON description presented previously.

![Figure 1: Context Model](image)

The following excerpts from a JSON/BSON document illustrate the description of an user/application, regarding time, location, Network, QoS, QoE, and QoD.

Table 1: MongoDB view of a current representation

```
"Location": {
    "latitude": -12.9822574,
    "longitude": -38.4586363,
    "altitude": "none",
    "atm_pressure": "none"
},

"Network": {
    "src_ip": "192.168.0.21",
    "flow_id": 17,
    "dest_port": "16000",
    "dest_port_low": "16000",
    "dest_port_high": "16038",
    "src_port": "16000",
    "src_port_low": "16000",
    "src_port_high": "16038",
    "dest_ip": "192.168.0.6"
},

"QoE": {
    "MOS": 4.409258999999997,
    "R-factor": 93.1999999999997,
    "time": "2015-05-18 15:03:52.632",
    "User_Percpetion": {
        "urgency": "false",
        "Mood": "false",
        "Back_Noise": "false",
        "MOS": 4
    }
},

"QoD": {
    "has_gps": "false",
    "gps_precision": "none",
    "process_cores": "two",
    "screen_resolution": "none",
    "processor_overload": "false",
    "resource_fault": "false"
},

"QoS": {
    "p_loss": 0,
    "round_trip_time": 93.1999999999997,
    "jitter": 3.420933333333327
}
```
CONTEXT-AWARE ADAPTIVE ROUTING FRAMEWORK

After having introduced the context model to be applied and the format of the information to be exchanged within the proposed architecture, it is possible to understand how the contextual information is processed and forwarding updates affect the global performance of the network. The architecture of the proposed system was conceived based on the description of two main functional groups:

- **Context Management Modules** – which are responsible for the analysis and filtering of contextual information, and;
- **Forwarding Management Modules** – which are responsible for processing contextual information and for the application forwarding rules on network switches.

The main functionalities of the proposed architecture are:

- To collect and share contextual information among network devices;
- To centralize storage of contextual information;
- To carry out validation and assessment of contextual information based on Quality of Context policies, and;
- To support the query and utilization of contextual information in order to update routing for context-sensitive services.

The proposed architecture called Context-Aware Adaptative Routing Framework (CAARF) was conceived based on the integration of different functional modules, as depicted in Figure 2:

- **Context Agent**: Responsible to receive contextual information from active communication devices, applications and users. These information is are forwarded to the context management module;
- **Context Handler**: Responsible to receive context information, record them on the context database within context model management module;
- **Context Model Management**: Responsible for storing contextual information, processing quality of context and making available information to context-based forwarding management module;
- **Context-based forwarding management**: Responsible for processing forwarding rules based on contextual information, and;
- **Flow adaptation**: Responsible for the notification of updated forwarding information on the respective active switching/forwarding communication device.

- **Context Agent**: Responsible to receive contextual information from active communication devices, applications and users. These information is are forwarded to the context management module;
- **Context Handler**: Responsible to receive context information, record them on the context database within context model management module;
- **Context Model Management**: Responsible for storing contextual information, processing quality of context and making available information to context-based forwarding management module;
- **Context-based forwarding management**: Responsible for processing forwarding rules based on contextual information, and;
- **Flow adaptation**: Responsible for the notification of updated forwarding information on the respective active switching/forwarding communication device.

Next section illustrates the implementation aspects of the proposed architecture.
CAARF: IMPLEMENTATION ASPECTS

In order to illustrate the operational and implementation aspects of the proposed architecture, consider the scenario depicted in Figure 3. As a case study, a pair of IP PBX, compatible to the SIP protocol, located in different networks was implemented. The traffic between both PBXs should pass through a common gateway, which supports traffic shaping. This is one of the cases supported by the Flow Adaptation module. Flow Adaptation supports a number of technologies and protocols such as MPLS, G MPLS, SPB, SDN-Like, Openflow and IP. In this work, we propose the implementation of traffic shaping using Netfilter (Iptables), as well as the utilization of queueing mechanisms and traffic control, Ibt and tc.

A SIP calling simulation software, Startrinity (Startrinity, 2015), has been applied in order to carry out simultaneous calls from an UAC (User Agent Client) to an IP PBX, in both directions, and complementarily to another SIP Tester Startrinity, as in the role of UAS (User Agent Server).

Initially, the data traffic generated and delivered without interruption will be measured with a respective MOS (Mean Opinion Score), which is related to R-factor, according to ITU-T rec G.107 specifications (Carvalho et al. 2005), meaning that each impairment on voice calls may be computed independently in order to achieve a score result. MOS is a score defined in a grade scale that starts from 1 (Not Acceptable) through 5 (Excellent). Therefore, MOS, will be collected by a Context Handler from data acquired by the Middleware Context Agent. At first, a good MOS reading will not affect routing decisions.

After some time, noisy traffic is generated, such as upstream and downstream traffic related to other concurrent traffic. This noisy traffic will lead to resources shortage, consequently affecting VoIP traffic with MOS violation against the agreed Service Level Agreement (SLA).

This violation should be processed accordingly in order to adapt routing. In order to better understand how the proposed architecture reacts to this scenario, the following events are numbered from 1 through 13:

1. **Middleware Context agents** collect periodically the contextual information from applications and network, such as location, timing, QoS, QoE and QoD, among other parameters;

2. **Context Middleware** prepares JSON contextual information to **context handler**, which will listen to the changes;

3. **Buffer management**, registers all information received in **context database** (repository located within **context model management**);

4. **Notification scheduler**, submodule of **context handler**, notifies **context model management** about the reception of incoming contextual information;

5. **Context Processing**, submodule of **context model management**, receives notification of new contextual information, retrieves information from **context database**, and submit them to the module **Quality of Context (QoC) verification**, which applies pre-defined QoC rules;

6. **Context processing**, submodule of **context model management**, registers a new context within **global context model**;

7. **Context processing**, submodule of **context model management**, notifies the module **context-based forwarding management** the context modification and QoE violations;

8. **Forwarding rules processing**, submodule of **context-based forwarding management**, receives notification
of new context and the results of queries to the global context model;

9. **Forwarding rules verification**, submodule of context-based forwarding management, queries global context model and applies pre-defined forwarding rules;

10. **Forwarding rules processing**, submodule of context-based forwarding management, records updated flows within flow repository;

11. **Forwarding rules processing**, submodule of context-based forwarding management, notifies flow adaption about the available updated flow to be applied on the routers;

12. **Flow adaption** verifies new flow rules on the current flow view, repository within context-based forwarding management, and;

13. **Flow adaption** applies new rules on network routers.

**CASE STUDY**

Considering the scenario depicted in Figure 3, the context database was implemented using the NoSQL MongoDB. Indeed, given the variety of existing context data, situations and context applicable, the description of a context with MongoDB turned out to be more effective in order to describe the extensive number of states and variables that the system should support. Moreover, MongoDB still provides a number of functions in order to describe queues, priority queues and stacks, which allow for an optimized context management and signaling between modules.

The developed system counts with a listening module (event handler) that is responsible for collecting data managed by the Middleware, which is already described using JSON/BSON and its respective MongoDB format. The QoC module is responsible for filtering the QoC events, discarding non-conformant data according to the system’s parameters or to the SLA agreement. For instance, it means outdated data, inaccurate data read from devices, SLA non-conformant data, inconsistent ports, divergent data (or out of scope concerning QoS, QoD), missing data, etc.

As an outcome of the filtering process, besides eliminating the detected inconsistencies, it still preserves the stability of the system, resulting in an event log database that will be processed by a system management module. At last, in this case study, oriented to converging applications a correlation between user/application QoE and SLA for a given service is also carried out.

Context Handler/Management processes the QoC rules as well QoE measurements validation against pre-defined set of user/application definitions, followed by a notification step, in the form of a QoE violation notification queue (Context Model Management → Forwarding Approach). This alert initiates a Flow Adaption routing decision reasoning.

In order to acquire data related to the collecting parameters, some monitoring and evaluation tools are employed. For this purpose, the Middleware Context Agent is responsible to interact with different available tools, being an external layer to the Context Controller that is in charge to collect data from sensors, probing and different measurement tools from multiple vendors.

The main advantage of this approach is the possibility to interact with different tools available among the community without the need to develop new collecting modules from scratch.

The calling Simulator, Startrinity, is able to generate SIP traffic logs and to provide parameters measurements for MOS, R-factor and other QoS parameters, such as jitter, packet loss and RTT. These parameters can be collected by the Handler, resulting in data already formatted by the Middleware collect agent, such as JSON format. In turn, the MongoDB database is made available on the cloud using the MongoLab (MongoLab, 2015) platform.

As a consequence, notifications are generated on the Forwarding module, and based on the routes evaluation map, which in this study case are implemented as forwarding queues according to the Netfilter definitions, and also based on the QoS records for that particular flow, the Forwarding module will modify the forwarding table. These modifications are also recorded on the events log, which will be useful for further generation of flow events reports.

In this experiment, successive rounds comprising fixed blocks of calls between the hosts configured as UAC (caller furing) and UAS (IP PBX role), have been performed. Bandwidth constraints are implemented through HTB rules applied to a Netfilter host, whose queues are defined as a sequence of step values ranging from (256Kbps, 700Kbps, 1400Kbps up to 2400kbp). Concurrent calls are then established in progressive scale, comprising a sequence [3,6,12,12,20] of simultaneous calls at each round, providing the system with the context data that will be processed by the Context Controller. Experiment rounds and queues changes in response to notifications issued against the Forwarding Module are shown in Table 2, and are related with MOS behavior according to Figure 5.

<table>
<thead>
<tr>
<th>Round</th>
<th>concurrent calls</th>
<th>Queue</th>
<th>BW, Kbps</th>
<th>Notify FWQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3</td>
<td>1.20</td>
<td>300</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>6</td>
<td>1.20</td>
<td>300</td>
<td>Y</td>
</tr>
<tr>
<td>3</td>
<td>6</td>
<td>1.40</td>
<td>700</td>
<td>Y</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>1.40</td>
<td>700</td>
<td>N</td>
</tr>
<tr>
<td>5</td>
<td>12</td>
<td>1.50</td>
<td>1400</td>
<td>N</td>
</tr>
<tr>
<td>6</td>
<td>20</td>
<td>1.50</td>
<td>1400</td>
<td>Y</td>
</tr>
<tr>
<td>7</td>
<td>20</td>
<td>1.60</td>
<td>2400</td>
<td>N</td>
</tr>
</tbody>
</table>

Context Handler acquires the data in each round and constantly updates the context database, located in MongoLab, as illustrated in another series of calls in Figure 4, while Context Model Management routines constantly assess the QoC and QoE status in order to meet SLA requirements. Successive QoE updates herein being understood as MOS computed values can trigger
notifications towards Context Forwarding Module in the case of requirements violation, shown as a blue arrow in Figure 4.

These notifications will eventually generate adjustments through computing of the necessary bandwidth required by the new context and leading to the choice of a new queue applied on Netfilter/iptables. In Figure 4 sequential readings that represent the context data are presented as well part of the MongoDB samplings acquired from four different states associated with their respective time stamps.

Context Controller in order to recover the MOS based on SLA.

Figure 5: MOS adaptation along successive SIP calls

CONCLUSIONS

This paper discussed the effectiveness of Quality of Experience (QoE) techniques in approach as an optimized solution for routing time-sensitive traffic regarding converged networks. In order to propose and focus on a user-centered solution, a solution that is capable of QoE evaluation has been proposed for the context-aware architecture.

The context model adopted is based on the description of the user’s quality of experience, quality of the device and quality of the infrastructure. The main goal is to propose a data model being scalable, flexible and generic and the JSON/BSON representation through MongoDB has been demonstrated as a powerful and flexible architecture that can lead to an extensive representation. Meanwhile, the proposed architecture provides main modules for monitoring and collecting user and network’s status, processing the changes in the status and proposing optimal routes based on the current status as well historical data, usage profile and behavior descriptions.

This work has been validated through a testbed that is capable of inferring results for different stress scenarios. As for future works, the proposed context model and architecture should be validated through the execution of different and more complex scenarios and context-aware update and routing optimization protocols, taking in account, as well, the user perception data acquired from different applications usages. A QoC monitoring approach is also under development, meaning that more diverse results and scenarios will be addressed in future works built upon the present framework.

ACKNOWLEDGEMENTS

This project has been supported by FCT – Fundação para a Ciência e Tecnologia, within the Project Scope: UID/EEA/50009/2013.

REFERENCES


---

**WEB REFERENCES**


THE NEED OF SIMULATION IN QUEUING MODELS FOR PORT LOGISTICS

Pasquale Legato
Rina M. Mazza
Department of Informatics, Modeling, Electronics and System Engineering
University of Calabria
87036 Rende CS, Italy
E-mail: legato@dimes.unical.it

KEYWORDS
Queuing, Discrete simulation, Performance analysis, Decision support systems.

ABSTRACT
Container discharge/loading and internal transfer between a containership and the storage/retrieval area on the yard of a maritime container terminal is a complex logistic process. Each container crosses all three subsystems of the terminal (i.e. quay, transfer and yard areas) and, while doing so, requires seizing and releasing both handling and transfer equipment (i.e. cranes and vehicles). This paper focuses on the integrated container transfer and storage/retrieval subprocess occurring on the yard subsystem. The objective is to assess the appropriate methodology for a reliable quantitative evaluation of some performance metrics. To this purpose, it is shown through numerical experiments based on real data that analytical queuing models are still not mature enough to replace discrete-event simulation in real-life environments.

INTRODUCTION AND PROBLEM STATEMENT

It is well stated by terminal operators that the overall productivity and quality of service of a terminal mainly depend on the efficient interaction of its major subsystems: the quay, transfer and yard areas. As illustrated in Figure 1, containers flow through these subsystems waiting to be discharged/loaded (D/L) by cranes working on berthed vessels (Quayside), transferred by vehicles between the quay and the yard according to specific routes (Transfer) and stacked/retrieved in the storage area (Yardside) by the proper handling equipment.

![Figure 1: Major Subsystems in a Container Terminal](image)

Since container flows are triggered by vessel arrivals that, in turn, can be controlled only to a very limited extent, terminal managers ask for an efficient organization and management of the entire internal container handling process (Stahlbeck and Voß, 2008; Angeloudis and Bell 2011). The occurrence of both crane starvation (during container loading) and crane blocking (during container discharge) is expected to be minimized, as a result of the application of mathematical models meant to support decision making. For performance evaluation and optimization under the dynamic allocation of resources in conjunction with the stochastic duration of logistic activities and unavoidable waiting phenomena, queuing models should be pursued, despite of the scarce attention received (Carlo et al. 2014a; Carlo et al. 2014b). Specifically, a queuing model of the container transfer and storage/retrieval process faces two main difficulties: the first arises in the representation of the waiting phenomena suffered by vehicles performing container transfer within the yard; the second arises in providing a reliable representation of the vehicle service process at the storage point within the yard. Both of the above difficulties are highlighted in the following.

As in most cases, from an organizational point of view, the yard is divided into large areas called zones. In each zone, containers are stacked into blocks. A block has: a number of lanes or rows ranging from 6 to 13 placed side by side; usually up to five containers in height called tiers for each row stack; 15 or more containers in length. A vertical section of a block is normally referred to as bay. Here, we consider a maritime container terminal devoted to pure transshipment in which the yard features a similar organization but with an “extensive”, rather than “intensive” layout. By this we mean that the yard is characterized by a great number of container blocks, each bearing a large number of rows (e.g. 32) and a limited stack height (e.g. 3) because of the available yard technology. Specifically, container yard technology is given by two different sizes of straddle carriers (SCs) that can provide for both container transfer between the quay and yard areas and container stacking (1-over-2 or 1-over-3).

During the discharge/loading process, an SC is assigned the task to repeatedly retrieve or stack containers from or to target positions on the yard. Whatever be the actual location of the container within the stack (i.e. ground, intermediate or top slot), handling and transfer operations within a given yard row require accessing that row in compliance with security measures that depend on the port of reference (for us Gioia Tauro, Italy). In order for access to be granted, no other SCs must already be performing operations in that row, nor in the adjacent ones. The simple example portrayed by Figure 2 illustrates how service start in the central row (i.e. SC entering the central row) issues a locking condition in the two adjacent rows, thus preventing the possibility of parallel
services neither in the left, nor in the right adjacent row for the whole duration of the service.

![Diagram](image)

**Figure 2: Service Locking among Yard Rows**

As a result, an SC can wait before gaining access to its target row not only because of the standard condition that the row is already occupied by another SC, but also because at least one of the two rows adjacent to the target one is already occupied. The latter waiting condition is not similar to the former because it negatively affects row utilization and throughput and causes a further increase of the number of SCs waiting in front of the row and, consequently, the waiting time of each queued SC.

The rest of the paper is organized as follows. Section 2 discusses the limits of an analytical solution for the queuing network addressed in this work. Section 3 focuses on the simulation of the transfer and storage/retrieval subsystems. Numerical experiments are presented in Section 4 and, finally, conclusions are drawn in Section 5.

**ANALYTICAL SOLUTION OF THE QUEUING SUBMODELS**

In our conceptual model in Figure 3, the queuing network model obtained by connecting the three queuing submodels previously discussed is a closed-type network model. This is because, in our view, a fixed-size fleet of vehicles is set as the fixed population of customers circulating among the above submodels. Containers, in turn, are viewed as passive resources that need to be acquired by vehicles from vessels berthed along the quay or from storage rows within the yard. Arriving and departing vessels determine the volume of containers to be handled by the fleet of vehicle-customers for which the optimal size could be determined by a "what-if" analysis supported by the queuing network model at hand.

![Diagram](image)

**Figure 3: Closed-type Queuing Network Model of the Terminal Subsystems**

This stated, the analytical solution of the queuing network model should be necessarily based upon the exploitation of the celebrated MVA (mean value analysis) algorithm (Reiser and Lavenberg, 1980), due to its major capability of incorporating variants to the basic schema and, thus, capturing realistic model features. We expect that a suitable variant to the central “delay equation” in the MVA could be successfully adopted to represent discharge/loading times under non-exponential distribution functions. The delay equation is used to evaluate the (average) customer waiting time in queue under the assumption of exponential service times (coefficient of variation = 1). On the other hand, we estimated from real data (courtesy of MCT SpA, Gioia Tauro, Italy) that the coefficient of variation from discharge/loading time ranges from 1 to 2 depending on the length and rarity of possible interruptions during normal operations. So, within the above range, an effective modification of the delay equation in the MVA can be based on the classical paradox of residual life (Kleinrock 1974). This modification has been numerically assessed against simulation experiments: results on average queue length and customer waiting time are usually within 10-15%, provided that the coefficient of variation is greater than 2 (Legato and Mazza 2015).

As for the analytical queuing representation of transfer times, it is worth mentioning that in the flat “extensive” yard we refer to, the actual congestion along the route followed by vehicles is rather limited. Figure 4 shows the behavior of the average waiting time at road intersection points as the number of SCs grows in three different “dummy” scenarios (i.e., A, B and C in which the average distance to be covered by an SC is different). In all three cases the waiting times grow significantly only with a relatively large number of SCs circulating on the roadgrid (≥5 SC assigned per quay crane when company practice is usually based on assigning 3 dedicated SCs per quay crane).

![Diagram](image)

**Figure 4: Waiting Time at Road Intersection Points during Container Transfer**

So, under similar (fortunate) conditions, we are confident that a service station with a number of infinite servers (IS) corresponding to the number of circulating vehicles is appropriate. This IS device allows to capture, as a whole, both the pure travel time and the stopping time at road intersections by means of a pure average service time under a given number of circulating vehicles. Fortunately, according to the BCCM theorem (Baskett et al. 1975) in queuing theory, the shape of the above service distribution does not
affect the applicability of the MVA. However, it is also worth mentioning that, as congestion along the routes increases, dependencies among the server-vehicles at the IS station become no longer negligible and, therefore, one of the classic obstacles in the practical application of analytical queuing models arises, i.e. dependency among servers.

Finally, coming to the problem of capturing the locking phenomenon among a given yard row and its adjacent ones, again we encounter a dependency phenomenon that is even worse than the dependency among servers: dependency among stations. The prerequisite of independency and station local balance (op. cit.), around which the analytical solution of queuing network models is developed, clashes with the phenomenon of service prevention at a given station due to the service start condition at another station. For this reason, the only possible analytical quantitative evaluation of the queuing network at hand can be carried out by relaxing the locking condition or, in other words, assuming that the workload imposed to any given yard block is so low that the locking condition is unlikely to occur.

**SIMULATION OF THE TRANSFER AND STORAGE/RETRIEVAL SUBSYSTEMS**

The limits of an analytical solution for the queuing network previously addressed leads us to solve it by using discrete-event simulation.

Starting from the general information provided in the first section, the integrated transfer and storage/retrieval subprocess is now tailored to mirror the organization and infrastructures of the transshipment container terminal located at the port in Gioia Tauro, Italy. Representing the highly-detailed container flow in this integrated model is the latest accomplishment in a long-lasting simulation-based decision support system designed and developed in cooperation with the terminal operator in Gioia Tauro (Canonaco et al. 2007; Legato et al. 2013).

In the transfer subsystem the vehicle-driver couple is modeled individually and the driver’s behavior is based on a classic (discrete) vehicle-following logic (Gazis et al. 1959). During actual container transportation, drivers move according to an en route rationale, meaning that they are provided with travel-related information after they start their trip (Chowdhury and Sadek 2003). In their step-by-step path choice, SC drivers move to the nearest intersection. If the intersection is not free, they queue before the intersection point until it becomes available for vehicle crossing (i.e. no gap acceptance). They then choose a path according to the traffic level along the alternative vertical/horizontal corridors where no overtakings occur.

In the storage/retrieval subsystem, once on target, SCs seek access to the rows in which container handling operations are to be performed. If the target rows are occupied or locked, SCs queue in front of them and wait for their turn according to a FIFO discipline. Operations duration in the yard row include the local motion time required to reach the container stacks of interest, as well as container stacking/retrieval times. In retrieval operations, additional time may be required to perform container reshuffling if other containers are stacked above the containers to be handled (Legato and Mazza 2013).

Technically speaking, the simulation model is developed according to an event-scheduling world view (Pidd 2004). The primary events occurring within the transfer and storage/retrieval submodel are listed in Table 1.

<table>
<thead>
<tr>
<th>N</th>
<th>Event</th>
<th>Resources</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>SCintersection (2)</td>
<td>intersection travel</td>
</tr>
<tr>
<td>2</td>
<td>SCsegment travel (1&amp;3)</td>
<td>travel intersection</td>
</tr>
<tr>
<td>3</td>
<td>SCarrival row (4)</td>
<td>row travel</td>
</tr>
<tr>
<td>4</td>
<td>SCdeparture from row (2)</td>
<td>travel row</td>
</tr>
</tbody>
</table>

Every event calls for the scheduling of specific activities, and requires seizing and/or releasing resources. Events also call other events (numbered in parenthesis). With respect to the main function of the submodel, the SCarrival row and SCdeparture from row events are of particular interest. They trigger the estimation of the following quantitative metrics:

- probability of SCs waiting before yard rows because of occupied rows;
- probability of SCs waiting before yard rows because of locked rows;
- SC waiting time before yard rows;
- container stacking/retrieval time;
- yard row throughput.

**NUMERICAL EXPERIMENTS**

The simulation model previously described has been designed and developed in compliance with all the conventional steps used to guide a thorough and sound simulation study (Banks et al. 2001). Microsoft Visual Basic 6.0 Professional has been used for the application development and all numerical experiments have been run on a personal computer equipped with a 2.26GHz Inter CoreTM2 duo processor and 2.93 GB of RAM.

Whatever be the scenario to simulate, the related input is defined by using a simple GUI panel in which the number and physical features of both terminal infrastructure (i.e. berth and yard) and equipment (i.e. quay cranes and SCs) are specified. For the sake of simplicity, here we consider only a container discharge scenario whose general settings are defined in Table 2.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Number</th>
<th>Characteristics</th>
</tr>
</thead>
<tbody>
<tr>
<td>berth</td>
<td>20 bollards</td>
<td>24 meters per bollard</td>
</tr>
<tr>
<td>yard</td>
<td>3 blocks/pa</td>
<td>32 meters and 16 bays per block</td>
</tr>
<tr>
<td>QCs</td>
<td>8 (rail)</td>
<td>12 six-hour work shifts</td>
</tr>
<tr>
<td>SCs</td>
<td>3 per QC</td>
<td>4 m/s loaded, 5 m/s unloaded</td>
</tr>
</tbody>
</table>

The object of this scenario is to analyze both yard and vehicle performance with respect to the waiting and locking
phenomena occurring on the storage area when all the containers discharged from a vessel are meant to be stacked in the same yard block. During transshipment from one vessel to another, container transfer and storage in the same block is quite common for those containers bearing the same port of destination. To fix ideas, let B2 in Figure 5 be the target block to which containers are delivered by (24) SCs after discharge operations provided by (8) quay cranes QC1, QC2, QC3, QC4, QC5, QC6, QC7 and QC8 located at bollards 4, 6, 8, 10, 13, 16, 18 and 20, respectively.

The above so-called dependency among stations is very important. Similar to the (average) container transfer time, the waiting probability WP of an SC at a given intersection and the corresponding average waiting time WT (in seconds) do not depend on whether or not the queuing network model accounts for dependent row stations. As expected, Table 3 confirms that SC congestion at intersections (Int No.) of the roadway grid is only slightly affected by dependency among stations, provided that the queue length in front of the yard rows is rather limited.

<table>
<thead>
<tr>
<th>Int No.</th>
<th>Station Dependency</th>
<th>No Station Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>WP</td>
<td>WT</td>
</tr>
<tr>
<td>6</td>
<td>0.23</td>
<td>2.76</td>
</tr>
<tr>
<td>7</td>
<td>0.20</td>
<td>2.31</td>
</tr>
<tr>
<td>8</td>
<td>0.12</td>
<td>2.23</td>
</tr>
<tr>
<td>11</td>
<td>0.25</td>
<td>2.89</td>
</tr>
<tr>
<td>12</td>
<td>0.20</td>
<td>2.27</td>
</tr>
<tr>
<td>13</td>
<td>0.11</td>
<td>2.20</td>
</tr>
</tbody>
</table>

On the other hand, the waiting probability of an SC at a target row P(SC waits) and the corresponding average waiting time WT (in seconds) are seriously affected by dependency among row stations. As one may see in Table 4, relaxing the locking condition in dependent rows means estimating P(SC waits) to be equal to 0.17 when, instead, it is almost three times that value (i.e. 0.47). It also means downsizing the average waiting time of the SC by 27% (i.e. 90.6 vs 65.9). Both of these underestimations are due to the fact that in the presence of dependent stations, if an SC waits before a yard row, 37% of the times it waits because the row is occupied, whereas in the remaining 63% of the cases it waits because the row is locked. As a result, in this specific case, the locking phenomenon is not negligible.

<table>
<thead>
<tr>
<th>Performance</th>
<th>Station dependency</th>
<th>No Station Dependency</th>
</tr>
</thead>
<tbody>
<tr>
<td>P(SC waits)</td>
<td>0.47</td>
<td>0.17</td>
</tr>
<tr>
<td>Average WT (secs)</td>
<td>90.6</td>
<td>65.9</td>
</tr>
<tr>
<td>P(row occupiedISC waits)</td>
<td>0.37</td>
<td>0.17</td>
</tr>
<tr>
<td>P(row lockedISC waits)</td>
<td>0.63</td>
<td>-</td>
</tr>
</tbody>
</table>

**CONCLUSIONS**

The suitability of the analytical solution of a queuing network designed to model the main logistic processes in a maritime container terminal has been investigated in this paper. It is possible to get a reasonable accuracy in estimating average throughput and queue length based performance metrics under non exponential distributions for service durations. Greater and not yet overcome difficulties arise in the analytical modeling of realistic features such as vehicle congestion along the travel paths within the yard, as well as mutual service prevention phenomena due to the
interaction of storage/retrieval operations within adjacent storage rows. It has been shown through numerical experiments that dependency phenomena among servers and, even worse, among queuing stations still require the careful development and use of discrete-event simulation procedures for solving this kind of queuing network models. Hence, further work on refining methodologies for complex simulation conceptual models, as well as statistical analysis of simulation input and output are welcome.

REFERENCES


BIOGRAPHIES

PASQUALE LEGATO is an Associate Professor of Operations Research in the Department of Informatics, Modeling, Electronics and System Engineering at the University of Calabria, Rende (CS, Italy), where he teaches courses on simulation for system performance evaluation. He has published on queuing network models for job shop and logistic systems, as well as on integer programming models. He has been involved in several national and international applied research projects and is serving as a reviewer for many international journals. His current research activities focus on the development and analysis of queuing network models for logistic systems, discrete-event simulation and the integration of simulation output analysis techniques with combinatorial optimization algorithms for real-life applications. His email is legato@dimes.unical.it and his webpage is wwwinfo.dimes.unical.it/legato.

RINA MARY MAZZA is the Research Manager in the Department of Informatics, Modeling, Electronics and System Engineering at the University of Calabria, Rende (CS, Italy). She received a Ph.D. degree in Operations Research from the above university. She has a seven-year working experience on knowledge management and quality assurance in research centers. She is also a consultant for operations modeling and simulation in terminal containers. Her current research interests include discrete-event simulation and optimum-seeking by simulation in complex systems. Her e-mail address is rmazza@dimes.unical.it.
Impact of Different Input Data Sets on the Sorting System Performance of a Parcel Transshipment Terminal

Uwe Clausen  
Daniel Dickmann  
Ina Dormuth    
Institute of Transport Logistics  
TU Dortmund University  
Leonhard-Euler-Str. 2  
D-44227 Dortmund, Germany  
E-Mail: {clausen, diekmann, dormuth}@ilt.tu-dortmund.de

KEYWORDS  
Logistics, parcel delivery industry, transshipment terminal, input data, discrete-event simulation

ABSTRACT

Operators of parcel transshipment terminals face the challenge to quickly and efficiently sort and transfer a high amount of parcels. In order to meet customers’ expectations short sorting intervals are required. If parcels are not loaded on time they miss their onwards connections within the network. However, the amount and characteristics of incoming vehicles and parcels changes over time and affects the sorting system performance and sorting intervals. This paper focuses on the structure of input data sets of parcel transshipment terminals. Five different data sets are developed e.g. by using kernel density estimation. A discrete-event simulation model evaluates the impact of these input data sets on the sorting system performance.

INTRODUCTION

The parcel delivery industry (PDI) has become a significant segment of the transport industry. Market participants in the field have specialized in the universal delivery of mainly standardized low weight shipments. Several factors, such as the increasing demand for time-critical shipments, the trend towards smaller load units or the rapid rise of e-commerce, facilitated this development and resulted in a continuous growth of the PDI segment. Major players that operate in this very competitive market segment are UPS (United Parcel Service), FedEx, Deutsche Post DHL, TNT or DPD (Dynamic Parcel Distribution) (Kille and Schwenmer 2013).

To process the large amount, parcels service providers use a transportation network with several hubs and depots. These transshipment terminals serve as collection and distribution points. They unload, sort and load the parcels according to their dedicated destinations and are usually equipped with automatic sorting systems (Bloss 2013).

A challenge for terminal operators is the increasing amount of parcels. This affects the sorting system performance and sorting interval of a transshipment terminal. In addition to that, network schedules, parcel dimensions and also parcel amounts of onward connections are subjected to change. These changes should be carefully observed in future strategic planning scenarios and are investigated in this paper.

The objective of this paper is twofold. First we create a data concept for the input data generation of a parcel transshipment terminal and develop future scenarios. Afterwards, we evaluate the impact of different input data sets on the transshipment terminal performance by using a simulation model. Hence, the paper is organized as follows: In section 2 the concept of the data generator is described. Section 3 explains our experimentation plan and the way new data sets are generated. In section 4 the considered transshipment terminal is explained. Section 5 presents our results and section 6 concludes the main findings.

STRUCTURE OF INPUT DATA & PARAMETERS

The material flow of a parcel transshipment terminal consists of three main process steps. First, incoming vehicles are assigned to unloading docks and manually unloaded. Second, the parcels are sorted with an internal network of conveyors according to their dedicated destination. Finally, the parcels are manually loaded into outgoing vehicles. Accordingly, the input data consists of the three object types Incoming Vehicle, Parcel and Outgoing Vehicle as shown in Figure 1.

![Figure 1: Objects and Attributes of the Input Data](image)

Attributes of the Incoming Vehicle include the Vehicle ID and Vehicle Name to ensure a clear identification. Additionally, the Vehicle Type defines the sizes of a vehicle. In this approach, we defined four different vehicle types that are commonly used in practice. The first two types are swap bodies (1) and semitrailer (2) that usually operated between different transshipment terminals within the transportation network. For the local traffic small to mid-sized trucks (3) and delivery vehicles (4) are used. The Vehicle Type determines the (maximum) Number of Parcels. The last attribute is the Arrival Time of the vehicle.

The object Parcel is labeled Parcel ID for clear identification. It is linked by the Vehicle ID to the associated
vehicle. The attributes *Destination* defines the internal sorting destination of the vehicle. The attribute *Sorting Capability* states whether a parcel can be sorted with the automatic system. Parcels that do not meet the requirements of the automatic sorting system (e.g. due to form or shape) are called non-conveyable parcels and must be handled in a manual sorting system. The attribute *Machine Readability* defines whether it is a readable or a non-readable (no read)- parcel. Parcel service providers try to keep the percentage of non-readable as low as possible since additional handling affords occur to identify and label these shipments. The last three attribute *Parcel Length*, *Parcel Width* and *Parcel Height* define the dimension of the parcel.

Attributes of the outgoing vehicle are similar to those of the incoming vehicle (*Vehicle ID*, *Vehicle Name*, and *Vehicle Type*). The vehicle is linked by the attribute *Destination* with the associated parcels. The attribute *Capacity* defines the maximum capacity according to the vehicle type. Since our approach assumes that there are enough outgoing vehicles available, this work focuses on the input data objects *Incoming Vehicle* and *Parcel*.

Based on the input data concept, we analyzed the characteristics of the data set. Three relevant input parameters are identified which are suitable to describe the structure of an input data set. Those parameters are subjected to be changed in future scenarios and are visualized in Figure 2.

![Figure 2: Influencing Parameters](image)

The first parameter summarizes the arrival times of the incoming vehicle – hereinafter called *arrival pattern*. It describes the time stamps of incoming vehicles. The arrival pattern depends on the location of the transshipment terminal and its function within the transportation network. In our basic scenario we use an incoming vehicle arrival pattern derived from real data. In a future scenario (2a) we examine later arrival times of the incoming vehicles. Following the current development, customers are increasingly demanding late pickups, which results in later arrival times and cut-offs. Additionally, travel times are lengthened by the consolidation of transportation processes (Clausen und Goedicke 2012).

The second parameter focuses on the parcel characteristics. It summarizes attributes of the parcel dimensions (*Length, Width, and Height*), *Sorting Capability* and *Machine Readability*. The paper itself focuses on the parcel dimensions. In practice, there are significant differences between parcel service providers in terms of average parcel dimensions. Service providers that are mainly serving customers in the business-to-business (B2B) segment usually have a bigger parcel dimensions than those with a high business-to-consumer (B2C)-share. Due to the increasing share of smaller B2C-parcels the average overall parcel dimensions shrinks. Dullinger presents exemplary data for parcel length and expects that average length will drop from currently 530 mm to 430 mm (Dullinger 2008). In the basic scenario our shipment sizes are derived from a parcel service provider with a relatively high share of B2C-parcels. The future scenario (2b) represents a scenario with even smaller shipment sizes.

The last parameter summarizes the characteristics of the parcels destinations. This parameter depends on the location and the function of the transshipment within the network. In the basic scenario the amount of parcels varies from destination to destination. To generate the future scenario (2c), which contains larger parcel amounts for certain destinations, we used a fitting exponential distribution. As a result the destination distribution is less well-balanced. This concentration can be caused by different regional economic developments or a higher concentration of parcel receivers in city areas.

**EXPERIMENTATION PLAN AND GENERATION OF DATA**

Based on the above mentioned parameters the experimentation plan for the different input data sets, as shown in Figure 3, is created. The current input data set, derived from real data, is used as a starting point. It includes 230 vehicles with around 245,000 parcels. Due to the general market growth of the CEP industry, we expect a market growth of 15% in all future scenarios.

Therefore, our basic scenario (1) contains 264 trucks with around 282,000 parcels. In a next step the three different scenarios (2a), (2b) and (2c) are generated. Scenario (2a) includes 264 trucks with an average delay of 42 minutes. Scenario (2b) contains smaller parcels sizes. The amount of parcel remains the same. Accordingly, the number of incoming vehicles drops from 264 to 230 trucks. Scenario (2c) increases the imbalance of the parcels destination. The share of the top ten sorting destinations rises from 14% to 18%. Scenario (3) combines the delayed arrival time, the smaller parcels and the imbalanced distribution.

A way to generate arrival patterns is to use theoretical distributions (Sokolowski and Banks 2010). Therefore, the derived real arrival times of the incoming vehicles are compared with different probability density functions, e.g. the poisson distribution that is commonly used for arrival rates (Gallager 2012). As none of the selected functions provided a good fit, we decided to use a different solution to generate the delayed arrival pattern. The kernel density
estimation is an approach that does not require specific probabilistic assumptions (Wand and Jones 1995).

Together with this, the share of top ten relations increases from 14% to 18%.

**Figure 3: Experimental Plan**

The idea is that every data point can be regarded as probability mass. The parameter bandwidth defines the quality of approximation. For each observation the density consequently increases at the associated time slice. The left side of Figure 4 shows the arrival pattern of the basic scenario (dashed line represents the density of the vehicles). The right side visualizes the generated scenario (2a). The bandwidth = 30 is used as the smoothing parameter for both data sets. To compress the arrival function a triangle distribution is used. This function adds delay times between 0 and around 95 minutes to the vehicles. The last arriving vehicles have no or only a small delay in order to avoid arrival after their cutoffs.

**Figure 4: Arrival Pattern (Basic vs. Delayed Scenario)**

For the smaller parcels scenario (2b) lower parcel dimensions (length, width, and height) are generated randomly (Birta and Abrez 2007). This size reduction leads to an increasing amount of parcel per incoming vehicle. On average, the new data set contains 1,224 parcels per vehicle instead of 1,067 parcels in the basic scenario. The distribution of the parcel amount over the vehicles is shown in Figure 5.

In the new destination distribution scenario (2c) a higher amount of parcels is concentrated on around 30 destinations. The proportion for the destinations rises from 58% to 75%.

**Figure 5: Parcel Characteristics (Basic vs. Smaller Parcel Scenario)**

**SIMULATION MODEL**

To study the effects of the different input data sets a simulation model is created with the simulation software Enterprise Dynamics (ED). Used objects are based on the transportation logistics environment called ED Transport that was developed at the Institute of Transport Logistics. (Neumann and Deymann 2008; Clausen and Diekmann 2012)

The simulation model is shown in Figure 6. It represents a modern parcel distribution hub, reaching sorting capacities up to 35,000 parcels per hour.

**Figure 6: Screenshot of the Simulation Model**

The building is U-shaped. The building area covers around 24,000 m². In total, six unloading areas are located at the front docks, feeding two main sorters that transport the parcels to the loading area.

Incoming vehicles are assigned by First-Come-First-Serve-policy to the unloading docks. Vehicles that cannot be
served immediately are equally assigned to the six unloading areas. Inside these unloading areas, the vehicle with the highest amount of parcels will be allocated first. That assignment policy is similar to the longest processing time (LPT) rule which is used for scheduling algorithms (Pinedo 2005). The strategy makes sure that swap bodies with long unloading times (caused by many parcels) are not assigned at the end of an interval and therefore do not extend the sorting operations. The vehicle replacement time in our model is 3 minutes which is common for shunting swap bodies. Unloading rates are modelled as stochastic functions with parameters for automated bulk unloading technology.

The sorting system consists of two main loop sorters that transfer the shipments from the unloading to the loading area. Loading rates are stochastic. If the arrival rate of incoming parcel in a certain time period exceeds the unloading rate, the parcels are queued in the loading dock buffer. In case that a buffer reaches its maximum capacity, incoming parcels for this destination remain on the main sorters. These so called recirculating parcels reduce the sorting performance significantly.

The assignment strategy of outgoing destination to end points of the sorter aims at equally balancing workload over the main sorters sections. Hereby, it is avoided that too many parcels are transferred to the same section. This causes blockings and reduces the efficiency of the sorting system. Furthermore, highly frequecanted destinations are split into two or more destinations to reduce buffer overflows and recirculation.

RESULTS & INTERPRETATION

In this section the simulation results are presented. In order to analyze the impact of our input data we compare the result of the basis scenario (1) with the considered scenario (2a), (2b), (2c) and (3). A standard output sheet that summarizes the main results has been developed. These sheets are presented for each comparison. The make span (MSP) describes the time of sorting and is shown on the horizontal x-axis. The interval between the first and the last time stamp is marked with a black double arrow. The grey background represents the sorter throughput rate per time slice (5 minutes). The amount of sorted parcels per time slice is counted and displayed on the right hand y-axis. In addition to that, we measure the workload balance (WLB). It is defined as the share of each sorter section over time and is illustrated on the left hand y-axis. Since our transshipment terminal has four sorter sections, the equal (balanced) share is 25%. Our findings of each comparison are discussed in the following.

Results of scenario 2a (Modified arrival pattern)

As shown in Figure 7 the delayed arrival of all trucks causes a lower sorting throughput especially at the beginning of the time interval. This lower efficiency can be explained with less available trucks (7 instead of 53) from the start. Consequently, the MSP of 112 time slices is higher compared to 110 slices in the basic scenario. The WLB follows in both scenarios a very similar pattern. All four sorting sections are quite equally balanced around a share of 25%.

Figure 7: Results of the Simulation Model for Basic and Arrival Pattern Scenario

Results of scenario 2b (Modified parcel characteristics)

Figure 8 shows the results of the modified parcel characteristics. The overall MSP is reduced from 110 to 106 time slices. This increased efficiency is caused by fewer trucks and consequently less trailer replacement times. Due to this, a higher sorting throughput rate can be identified during the whole sorting interval. WLB of modified parcel characteristics are slightly less imbalanced compared to the basic scenario.

Figure 8: Results of the Simulation Model for Basic and Parcel Characteristics Scenario

Results of scenario 2c (Modified destination distribution)

In the destination distribution scenario a higher number of parcels is concentrated on several relations. Consequently, the capacity of these destinations is often exceeded. The amount of recirculation parcels is high and negatively affects the sorter throughput rate. As it can be seen in Figure 9 the MSP increases from 110 to 122 time slices. The WLB is higher compared to the basic scenario.

Figure 9: Results of the Simulation Model for Basic and Destination Distribution Scenario
Results of scenario 3 (Combined scenario)

Figure 10 shows the results of the combined parameters scenario. The MSP increases from 110 to 127 time slices. The sorting interval increases to 85 minutes, resulting in a high amount of parcels that misses the onward connections. Especially the late arrival of trucks at the beginning of the interval in combination with the unequal distribution has a negative impact on the sorting system performance between 11 pm and 1 am.

<table>
<thead>
<tr>
<th>Scenario Basic (1)</th>
<th>Scenario Combined Parameters (3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MSP: 110</td>
<td>MSP: 127</td>
</tr>
</tbody>
</table>

Figure 10: Results of the Simulation Model for Basic and Combined Parameters Scenario

CONCLUSION AND FUTURE WORK

This paper evaluates the impact of different input data sets on the sorting system performance in the parcel delivery. Based on expected future developments we create five input data sets and compare their results. The delayed arrival scenario extends the sorting interval by about 10 minutes. Overall impact is very limited, as the majority of the delayed vehicles are generated at the beginning of the interval. The smaller parcel scenario reduces the sorting interval about 20 minutes due to reduced vehicle replacement. The modified destination distribution has a large impact on the sorting performance and extends the sorting interval about 60 minutes. A high amount of recirculating parcels reduces the sorting performance. The combined scenario results in a delay of 85 minutes. Future work will develop additional input data sets with combined parameter changes. Furthermore, extra work will be done to analyze the characteristics (e.g. correlations) and changes of real data in order to generate input data as realistic as possible.

ACKNOWLEDGEMENTS

The presented results are parts of the work from the research project „Development of a combined approach that links discrete mathematical optimization and stochastic simulation for planning and operating logistics nodes (applied to transshipment terminals in the parcel delivery industry)“ funded by the Deutsche Forschungsgemeinschaft (DFG - German Research Foundation).

REFERENCES


AUTHOR BIOGRAPHIES

UWE CLAUSEN is the managing director of the Institute of Transport Logistics at TU Dortmund University and director of the Fraunhofer-Institute for Material Flow and Logistics (IML). He studied computer sciences at the University of Karlsruhe (TH) and finished his doctoral thesis on transportation network optimization at TU Dortmund University in 1995. Afterwards, he worked at the “Deutsche Post AG” and joined Amazon.de in 1999. In 2001 Uwe Clausen accepted the chair “Transport Systems and Logistics” at TU Dortmund University, which became the Institute of Transport Logistics in 2011. He is amongst others chairman of EffizienzCluster LogistikRuhr and a member of ECTRI European Conference of Transport Research Institutes and of the Scientific Advisory Board of German Logistics Association (BVL).

DANIEL DIEKMANN attended TU Dortmund University, where he studied logistics and obtained his degree in 2010. He works at the Institute of Transport Logistics at TU Dortmund University as a scientific assistant and as a full-time PhD student, doing research in the fields of simulation in traffic and transport logistics.

INA DORMUTH attends TU Dortmund University, where she studies Statistics. She works at the Institute of Transport Logistics at TU Dortmund University as a student assistant, doing research in data modelling.
PICKUP AND DELIVERY SELECTION WITH A FIXED VEHICLE COST

Katrien Ramaekers
Gerrit K. Janssens
Tabitha Maes
An Caris

Research Group Logistics, Faculty of Business Economics
Universiteit Hasselt – campus Diepenbeek
Agoralaan 1, 3590 Diepenbeek, Belgium
E-mail: {katrien.ramaekers,an.caris,gerrit.janssens,tabitha.maes}@uhasselt.be

KEYWORDS
vehicle routing, pickup and delivery problem, meta-heuristics, optimisation

ABSTRACT

A pickup and delivery problem is a special case of the vehicle routing problem in which goods at customer sites are either picked up or delivered. A carrier has only a limited capacity within his own vehicle fleet. Therefore the carrier can only serve a selection of customers. Transport requests of clients are accepted only if they contribute to a higher total profit. A paired pickup and delivery selection problem is hardly investigated in literature. In case the carrier has a fixed fleet with a set of drivers, it is realistic that drivers have to be paid whether the truck is used or not. This leads to a fixed cost per vehicle in the decision problem. This practical aspect is modeled with the Pickup and Delivery with selection of customers. A mixed-integer programming formulation is given. A meta-heuristic method, more specifically a tabu-embedded simulated annealing algorithm, is developed to solve the problem in an efficient way. The heuristic is explained in detail.

INTRODUCTION

Shipping of goods involves several actors. In Maes et al. (2011) a conceptual framework is presented to model freight transport. The key actors in the framework include firms, carriers, and forwarders. Each day a carrier receives transport requests from his clients, which have to be executed within a certain time period. To obtain a maximal profit, the carrier groups certain orders and creates an optimal sequence of paired pickup and delivery tasks. In literature, this problem is called a pickup and delivery problem (PDP).

Within the PDP literature mostly it is assumed that all requests need to be fulfilled. In reality a carrier may refuse a transport request, which is believed to be profitable. If a request is accepted, it generates revenue when the transport is completed. A PDP in which a carrier has the decision power to accept a certain request or not, is known as a Pickup and Delivery Selection Problem (PDSP). This problem has been introduced by Schönberger et al. (2002) and by Schönberger (2005).

The request selection is evaluated by means of corresponding revenues and costs. The cost evaluation of a request requires the determination of a routing plan. The cost consists of a variable part related to the distance or time travelled. The variable part might be different depending on whether each transport request is fulfilled by the carrier or by a third party logistics service provider (LSP). Depending on the organisation of the carrier a fixed cost (for example, the labour cost of the drivers) is included or not. In other cases the customer requires fulfillment of the transport request by the carrier due to a specific contract. Such a request is called a 'compulsory request' (Ramaekers et al., 2015).

It is assumed that a carrier has a fixed amount of vehicles at his disposal to execute requests. This is a reasonable assumption when looking at short run operational decisions. Decisions on the number of vehicles in the fleet of the carrier are tactical decisions and the cost of the vehicles is considered in the long run. Furthermore, it is supposed that a carrier has to pay the drivers regardless whether they operate a vehicle or not, so no fixed vehicle cost is induced. Both Schönberger et al. (2002) and Arda et al. (2008) do not take into account a fixed vehicle cost. In this paper this assumption is relaxed. For each vehicle, executing a route, a fixed vehicle cost is added to the total operating cost of the carrier. The carrier has to consider whether or not a vehicle performs a route. Only when the profit obtained from a route is sufficiently high to cover the fixed vehicle cost the route is constructed. This might lead to accepting less transport requests from customers than the case without vehicle cost.

LITERATURE REVIEW

The PDP is a generalization of the vehicle routing problem (VRP) (Mitrovic-Minic, 1998). In a VRP generally all trip requests either originate or terminate at a single depot. In a PDP the trip requests are made between two locations that are outside the depot. In this section the division between paired and unpaired pickup and delivery points is used as in Parragh et al. (2008). Pickup and delivery vehicle routing problems are characterised with unpaired pickup and delivery locations. In this case an identical load is considered, and each unit picked up may be used to serve a delivery request. A classical pickup and delivery problem on the other hand, has paired pickup and
delivery locations. Every request is associated with a paired origin and destination location and a specified load.

The literature related to this specific case of the PDP, the pickup and delivery selection problem relies on two main bodies of routing literature. On the one hand the VRP with profits and on the other hand literature concerning PDP. A review of the literature on the VRP with profits and on the pickup and delivery selection problem has appeared in Ramaekers et al. (2015).

This paper offers the following novelties compared to existing research. The traditional PDP as a PDSP by allowing a selection of transportation requests. This leaves the carrier with the option to discard transportation requests which lead to a lower total profit. The problem at hand considers more than one commodity and paired pickup and delivery locations. This is different to the study of Ting and Liao (2012) where a single commodity is considered and pickup and delivery are unpaired. Furthermore, multiple vehicles are considered and transport loads are less-than-truckloads. In the study of Verweij and Aardal (2003) only a single vehicle is assumed and in the work of Arda et al. (2008), Frantzeskakis and Powel (1990) and Kleywegt and Papanastavou (1998) full truckloads are investigated.

The paired pickup and delivery locations, together with the multiple vehicles and less-than-truckload requests make the PDSP very hard to solve. The only paper that studies a PDSP with similar problem characteristics but in a different problem context is Schönberger et al. (2002). Their heuristic results are not compared to exact solutions or lower bounds and reported results are briefly described. This hinders the comparison of computational results.

PROBLEM FORMULATION

In this section a mathematical representation of the problem is given. First, the key characteristics of a PDSP are described. Next, all symbols are introduced. Finally the objective function and problem constraints are formulated. The problem is defined as a static PDSP problem. The formulation is an adaptation of the PDPTW formulation of Mitrovic-Minic (1998).

Key characteristics of a PDSP

Not all requests have to be accepted, but every fulfilled request leads to revenue. If a request is accepted, a reward is achieved when the transport is done successfully. For every request a hard time window is assigned to both the pickup and delivery location. A request is of the type less-than-truckload. Furthermore, pickup has to occur before delivery of each request (Precedence constraint) and pickup and delivery have to be performed by the same vehicle (Pairing constraint). In the model multiple vehicles are used of equal capacity. All vehicles depart from and return to a depot of the carrier. Finally, travel costs and travel times for each link are known and assumed to be constant.

Introduction of symbols

**Index Sets**

- \( i \in P = \{1, \ldots, n\} \): set of requests originating at the pickup locations.
- \( i \in N = P \cup D \cup O \): set of nodes where \( P = \{1, \ldots, n\} \) is the set of pickup locations, \( D = \{n+1, \ldots, 2n\} \) is the set of delivery locations, and \( O \) represents node 0, i.e. a single depot.
- \( k \in K = \{1, \ldots, |K|\} \): set of identical vehicles.

**Parameters**

- \( Q \): uniform vehicle capacity.
- \( ct \): unit distance travel cost which is the same for each vehicle.
- \( q_i \): the quantity to be picked up or delivered at node \( i \in N \setminus O \). It is positive if \( i \in P \) and negative if \( i \in D \).
- \( Rev_i \): the revenue earned from request \( i \in P \).
- \( d_{ij} \): the distance between two nodes \( i \) and \( j \in N \).
- \( t_i \): the travel time between two nodes \( i \) and \( j \in N \).
- \( s_{i} \): service time at node \( i \in N \setminus O \). By convention the service time \( s_{0} \) at the depot is zero.
- \( e_i \): earliest service start time at a node \( i \in N \setminus O \).
- \( l_i \): latest vehicle arrival time at a node \( i \in N \setminus O \).
- \( s_k \): the operations start time of a vehicle \( k \in K \).
- \( f_k \): the operations finish time of a vehicle \( k \in K \).

**Decision Variables**

Two sets of binary variables are defined: flow variables \( X \) and request acceptance variables \( Y \).

\[ X_{ij}^k = \begin{cases} 1 & \text{if vehicle } k \text{ travels from } i \text{ to } j; \ 0 & \text{otherwise} \end{cases} \quad (i,j \in N, k \in K).

\[ Y_i^k = \begin{cases} 1 & \text{if vehicle } k \text{ performs request } i; \ 0 & \text{otherwise} \end{cases} \quad (i \in P, k \in K). \]

Next to these binary variables, two sets of continuous variables are introduced to keep track of service completion times and vehicle loads.

\[ T_i = \text{service completion time at node } i \in N. \]

\[ L_k = \text{load of vehicle } k \text{ after serving node } i \in N. \]

**Requests**

A carrier receives a set \( P \) of requests. Each request consists of a pickup location, a delivery location, a quantity to be shipped and a revenue if the request is completely satisfied. The quantity \( q_i \) may either be a positive or negative number, depending on the type of operation, either a pickup or a delivery task.

**Locations**

Three different types of locations may be distinguished, each with their own time window. A set of pickup locations and a set of delivery locations are included, each with an earliest operation time \( e_i \), a latest operation time \( l_i \). A single depot \( O \) is available, where each vehicle starts and ends its route.

**Network**

A network \( G(A,N) \) is given, with \( N = P \cup D \cup O \) the set of nodes and \( A \) a set of undirected arcs. Within the network the distance between two nodes \( i \) and \( j \) is given as \( d_{ij} \). The travel cost \( ct \), expresses the charge for travelling a single distance unit. The cost to travel a link is expressed as

429
The last variable on the network is \( t_{ij} \) which represents the time needed to travel from node \( i \) to node \( j \).

**Vehicles** The carrier has a given homogenous fleet of own vehicles. Each vehicle \( k \) has a capacity \( Q \). Vehicles are bound in time by their driver, due to legal driving time restrictions. Each vehicle has a start time \( s_k \) and a finish time \( f_k \). To keep track of the content of the vehicle, so that it does not exceed the capacity, load variables \( L_i^k \) are introduced.

**Operations** A vehicle performs several operations on its route. Each pickup and delivery operation takes a certain amount of operation time \( o_{ij} \) to perform per unit that needs to be handled. The total time a vehicle spends at the pickup or delivery location is \( o_{ij} \cdot |q| \). A vehicle is allowed to arrive early at the location, but must then wait until the start of the time window. A vehicle is not allowed to arrive at a location after the time \( l_i \). A drive-first strategy (drive first and wait at the arrival location) is used for the PDSP.

**Objective function**

The objective of the PDSP is to maximize the profit collected along the vehicle tours. Profit is defined as the sum of the total revenue collected minus the total cost of performing the routes. The total revenue is found by accumulating all revenues of the requests that are accepted and executed.

\[
\text{Rev}_{\text{tot}} = \sum_{k \in K} \sum_{i \in P} \text{Rev}_i \cdot Y_i^k
\]

The total cost (\( C_{\text{tot}} \)) is calculated as the sum of the costs of each link travelled by a certain vehicle \( k \).

\[
C_{\text{tot}} = \sum_{k \in K} \sum_{i \in N} \sum_{j \in N} c_{ij} \cdot X_{ij}^k + \sum_{k \in K} \sum_{i \in N} c_{\text{veh}} \cdot X_{ij}^k
\]

The objective function to be maximized is:

\[
\text{Profit} = [\text{Rev}_{\text{tot}} - C_{\text{tot}}].
\]

**Constraints**

**Flow conservation constraints** These constraints are introduced to make sure that vehicles entering a location leave the location.

\[
\sum_{j=1 \atop j \neq i}^{N} X_i^k - \sum_{j=1}^{N} X_j^k = 0, \forall i \in N, \forall k \in K
\]

**Vehicle constraints** Each vehicle starts and ends its route at the depot. An unused vehicle stays at the depot. An unused vehicle stays at the depot.

\[
\sum_{j \in P} X_{ij}^k \leq 1, \forall k \in K
\]

\[
\sum_{i \in D} X_{i(2n+1)}^k \leq 1, \forall k \in K
\]

Every request may be executed by at most one vehicle.

\[
\sum_{k=1}^{K} Y_i^k \leq 1, \forall i \in P
\]

A vehicle cannot load more freight than its capacity.

\[
L_i^k \leq Q_{\text{max}}
\]

To keep track of the load of a vehicle at a certain moment, the following constraints are necessary. Each vehicle leaves from and returns to the depot empty.

\[
I_0^k = 0, \forall k \in K
\]

\[
t_i^k - L_i^k - |q_i| \geq M_1 \cdot (1 - X_{ij}^k), \forall i,j \in N \text{ and } i \neq j, \forall k \in K
\]

**Time window constraints**

Each node has to be served within its time window. The start of the operation, as well as the end of the operation has to fall within the time window.

\[
e_i + o_{ij} \cdot |q_i| \leq T_i^k \leq l_i, \forall i \in N \setminus 0, \forall k \in K
\]

To keep track of time, a time variable is introduced. Initially, the time variable is set equal to the start time of the vehicle.

\[
T_0^k = s_k, \forall k \in K
\]

A vehicle may not exceed its finish time.

\[
s_k \leq T_i^k \leq f_k, \forall i \in N, \forall k \in K
\]

The arrival time at a node may not precede the earliest operation time allowed on that location. This is specified in the following constraint:

\[
t_i^k + l_i - T_i^k + o_{ij} \cdot |q_i| \leq (1 - X_{ij}^k) \cdot M_2, \forall i,j \in N, \forall k \in K
\]

Due to the time window constraint on \( T_i^k \), it is assured that the operation does not start before \( e_i \).

**Pairing and precedence constraints**

If a request is performed, then vehicle \( k \) has to finish its operations at the pickup location \( i \) before it can visit the associated delivery location \( n+i \). This is known as the precedence constraint, expressed as:

\[
t_i^k + l_{i(n+i)} - T_{n+i}^k \leq (1 - X_{ij}^k) \cdot M_2, \forall i \in P, \forall k \in K
\]

It is not allowed to split a request over multiple vehicles. A vehicle has to perform both the pickup and the delivery operation. This is known as the pairing constraint, expressed as:

\[
\sum_{j \in N \setminus 0} X_{ij}^k = Y_i^k, \forall i \in P, \forall k \in K
\]

\[
\sum_{j \in N \setminus 0} X_{ij}^k = Y_i^k, \forall i \in P, \forall k \in K
\]

In the formulation two big \( M \)-values are used, where \( M \) stands for a sufficiently large number. The value of \( M_1 \) is set equal to the capacity of the vehicles, and the value of \( M_2 \) is set equal to the maximum length of a working day.

**SOLUTION METHOD**

The problem under study is solved by means of a metaheuristic. The heuristic is based on the tabu-embedded simulated annealing algorithm of Li and Lim (2001).
algorithm starts with an insertion heuristic to create a first feasible solution \( (S_{\text{init}}) \). This solution is further improved by an improvement heuristic. Instead of repeating the tabu search until the procedure terminates, it is restarted from the current best solution \( (S_{\text{local}}) \) after several iterations (STOP) without improvement. At the same time the global annealing temperature \( T \) is reset. After a number of restarts \( K \) without improvement the algorithm is terminated. The generation of new best solutions \( (S'_{\text{local}}) \) is done via a tabu-search algorithm. The TSA-algorithm is described below with some explanation below on the various steps.

**TSA algorithm**

Empty tabu set

Set \( g\text{NoLmpr} = 0 \)

Set \( S_{\text{best}} \) to initial solution from insertion heuristic

\( S_{\text{best}} = \text{PostOptimize}(S_{\text{best}}) \)

Add fixed vehicle cost to each route cost

\( S_{\text{best}} = \text{LocalSearch}(S_{\text{best}}) \)

while \( g\text{NoLmpr} < K \) do

\( S_{\text{best}} = S_{\text{best}} \)

if Profit\( (S'_{\text{best}}) > \) Profit\( (S_{\text{best}}) \) then

\( S_{\text{best}} = S'_{\text{best}} \) and \( g\text{NoLmpr} = 0 \)

else

\( g\text{NoLmpr} = g\text{NoLmpr} + 1 \)

reset \( T \) to \( T_0 \)

end if

end while

Remove unprofitable routes

Set \( g\text{NoLmpr} = 0 \)

while \( g\text{NoLmpr} < K \) do

\( S = S_{\text{best}} \)

if Profit\( (S'_{\text{best}}) > \) Profit\( (S_{\text{best}}) \) then

\( S_{\text{best}} = S'_{\text{best}} \) and \( g\text{NoLmpr} = 0 \)

else

\( g\text{NoLmpr} = g\text{NoLmpr} + 1 \)

reset \( T \) to \( T_0 \)

end if

end while

Remove unprofitable routes

\( S_{\text{best}} = \text{LocalSearch}(S_{\text{best}}) \)

output \( S_{\text{best}} \)

First, the insertion heuristic and post-optimisation operators are run. Afterwards the fixed cost of the vehicle is added to the cost of each of the constructed routes. In a third step local search is applied to obtain the most profitable routes given the fixed vehicle cost. Step four checks whether all routes are profitable. A non-profitable route is removed and the requests from these routes are stored in a list. If one of the initially constructed routes is not profitable, fewer vehicles will be used in the optimal solution. Next, the TSA algorithm is rerun taken the previous solution (with the unprofitable routes) removed as input. However, this time only the remaining vehicles are considered. For this reason, the requests removed from a deleted route together with the other non-served requests are considered for insertion in one of the remaining routes during the second run of the algorithm. As fewer routes are considered, the run time of the rerun is lower than the first run. Finally, the profitability of the routes is evaluated again and the local search heuristic is applied.

The TSA algorithm makes use of some procedures called PostOptimize, LocalSearch and TABU.

The PostOptimize procedure relates to a REORDER operator. Staring from a solution, the operator tries to lower costs by reordering the nodes within existing routes. As no new requests are added, the revenue remains the same. Hence to increase profit, the cost of performing a route should decrease, i.e. when the total distance, the only cost driving factor decreases. The operator tries to improve the original route by serving pickup points earlier in the route and delivery point later.

The LocalSearch procedure relates to an improvement heuristic in which four local operators are used to improve the initial solutions. Two operators are classical PDP operators. Two operators, called SHIFT and EXCHANGE, are similar to the local search operators of Li and Lim (2002). Both other operators are specifically designed for this heuristic. The INSERT operator tries to insert, from a list of non-served requests, to insert a request in a route in feasible way. If the insertion of a such a request leads to a higher profit for the route considered, the request is inserted in the position with the lowest cost. The SWITCH operator removes the request with the lowest profit from the selected route and replaces it without a non-served request. These requests are switched if it results in a higher total profit.

The TABU\( (S) \) procedure aims to find from an existing solution \( S \) a new local optimum \( (S'_{\text{local}}) \) with the help of the SHUFFLE \( (S) \) algorithm. A random feasible solution \( (S') \) is generated with the SHUFFLE\( (S) \) algorithm, which is not in the tabu list. This random solution \( (S') \) is further optimized by means of the LocalSearch function. This leads to a new local optimum \( (S'_{\text{local}}) \), which is compared to the current local optimum \( (S_{\text{local}}) \). If it performs better the new local optimum is stored in \( S_{\text{local}} \) and transferred as an output of the TABU\( (S) \) algorithm. If the current local optimum \( (S_{\text{local}}) \) is not improved the search continues with the new found local optimum \( (S'_{\text{local}}) \). The algorithm repeats itself until no improvement is found for a certain number of iterations (STOP). The pseudo-code of the Tabu Search procedure is given below.

**Tabu Search procedure TABU\( (S) \)**

Set \( S_{\text{local}} = S \)

Set \( g\text{NoLmpr} = 0 \)

while \( g\text{NoLmpr} < \) STOP do

\( S' = \text{SHUFFLE}(S) \), with \( S' \) not in tabu set

\( S'_{\text{local}} = \text{LocalSearch}(S') \)

\( S'_{\text{local}} = \text{PostOptimize}(S'_{\text{local}}) \)

if Profit\( (S'_{\text{local}}) > \) Profit\( (S_{\text{local}}) \) then

\( S_{\text{local}} = S'_{\text{local}} \) and \( g\text{NoLmpr} = 0 \)

else

\( g\text{NoLmpr} = g\text{NoLmpr} + 1 \)

end if

end while
CONCLUSIONS

A variant of the Pick-up and Delivery Vehicle Routing Problem is investigated, in which the carrier can make a selection of customers to be served, depending on which transport requests offer profit. The research formulates the optimization problem as a mixed-integer linear programming problem and develops a meta-heuristic to solve the problem.

ACKNOWLEDGEMENTS

This work is supported by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office (research project COMEX, Combinatorial Optimization: Metaheuristics & Exact Methods).

REFERENCES

USING AGENT-BASED SIMULATION TO INVESTIGATE DAILY ORDER VARIATION OF A B2B FRESH FOOD SUPPLIER

Ross Clement, Parminder Kang, Alistair Duffy
Lean Engineering Group, De Montfort University
Leicester LE1 9BH, UK
rclement@dmu.ac.uk, pkang@dmu.ac.uk, apd@dmu.ac.uk

Ashley Hopwell
Zest Foods LTD
Little Eaton
Derbyshire, UK
Ashley.hopwell@zestproduce.co.uk

KEYWORDS
Agent-Based Simulation, Demand Forecasting, Fresh Food Suppliers, Simulation Trace Analysis

ABSTRACT
Agent-based simulation has been used to simulate customers of a B2B fresh food supplier, in order to examine why total orders vary considerably on a day by day basis. Different types of virtual customers can be included in the simulation, ordering products using different strategies including their own demand prediction. This simulation suggests that customers changing the day of their order is the largest cause of daily order variance.

INTRODUCTION
Daily demand forecasting is a technically challenging problem, especially for chilled and fresh produce sector. Currently available demand planning strategies and methodologies do not address the increased complexity of the fresh produce supply chain. Demand forecasting methods exist, such as ARIMA (Autoregressive integrated moving average) and other methods (Kandanandonid, 2012). The research in this paper is part of a larger effort to improve day by day demand prediction in the fresh food sector for SME (Small and Medium Sized Enterprises).

CONTEXT
Agent-Based simulation has been used to investigate how random variations in the ordering patterns of individual customers affect the total daily orders received by a B2B fresh food supplier. In particular, whether small variations in individual customers order amounts and/or times result in large variations in total daily orders as an emergent property of the customer base. Accurate daily demand is very important for businesses dealing in perishable products. If more fresh stock than be sold is ordered, it may have to be disposed of, or as an example be used for animal fodder. If insufficient stock is ordered, then it might not be possible to fulfil customer orders. In a competitive marketplace, customer satisfaction and retention are vital. Accurate daily predictions would mean that the correct amount of product with a small safety margin, could be ordered, leading to reduced waste and cost, and higher customer satisfaction. Individual customer orders can vary in that a customer may order a different amount of the product, and/or the customer may choose to order on a different day or days of the week. Customers may start, stop, and restart ordering if they switch temporarily or permanently to another supplier. All of these result in total daily orders changing.

Agent-Based simulation has been used to investigate the emergent properties of customers with varying ordering strategies and usage patterns of stock. Agent-Based Simulation has allowed creation of customer bases containing customers with minor randomised variations in order preferences, but also entirely different ordering strategies. This matches the real world situation where businesses such as pubs and restaurants will consume more or less fresh food products depending on their own customer demand, while schools and nursing homes will use fresh food in a more predictable fashion. As well as modelling these variations directly, causes of these variations have been investigated. Customers who order fresh products such as milk may store the product themselves in a fridge, and may only reorder when the existing product is running low or has exceeded its use-by date. This ‘fridge as buffer’ effect, assuming some fluctuations of demand, can result in populations where the total daily order varies wildly, and is difficult to predict.

Interviews with demand planners (Garcia-Taylor et al. 2015) revealed a large number of potential factors that might influence consumer demand, and therefore customer (of the B2B company) demand. E.g. weather is reputed to have a large effect on the demand for certain foods, e.g. strawberries and salad ingredients. However, visualisations of customer behaviour (Clement et al, 2015) and correlation analysis of ordering patterns suggested that other factors such as customers being gained and lost, as well as variance in the day that customers choose to order were also very significant contribution in the difficulty in predicting daily demand. This research is intended to further understand the role of the many factors that affect customer ordering, and lead to difficulties in predicting daily demand.

There are numerous examples where simulation has provided an advantage to understand and improve real world systems, e.g. validation before an existing system is altered or a new system built, to reduce the chances of failure to meet specifications, to eliminate unforeseen bottlenecks and uncertainties, to prevent under or over-
utilization of resources, and to optimize system performance (Maria, 1997; Kang et al. 2010). This paper discusses an Agent Based Simulation approach to address the B2B fresh produce supply chain issues.

Compared to individual based modelling, agents are seen as autonomous and active (Macal & North, 2005). Therefore, from the implementation viewpoint agents have some characteristics, which allow them to behave in a unique manner (Becker et al. 2006). Agents are Identifiable, Goal Directed, Self-Directed, and show Flexibility.

We used ABS to simulate the a B2B fresh food supplier system in order to investigate the daily demand changes due to the uncertainties introduced by the customer buying preferences.

While it was possible to find many examples of the use of ABS in understanding business processes and customer behavior, no example could be found where ABS was used to investigate the causes of variability in demand in a B2B fresh food supply chain context.

In our business context, customers place orders with the wholesaler, orders are collated, and products purchased from the wholesaler’s sources to satisfy customer demand. Note that not all orders will have been received by the time of purchase meaning that demand prediction is needed.

Because demand prediction is imperfect there is the risk of not obtaining enough product and failing to satisfy customers, or ordering too much product which may be wasted. Our objective is to minimise stock held, while always satisfying customer demand. (Raju et. al, 2015).

**MODEL OVERVIEW**

The overall process modelled is that of a B2B fresh food supplier, referred to as the 'company'. The 'customers' of this company are other businesses that will then sell products to their own customers, whom we will describe as 'consumers' to avoid confusion. Similar B2B fresh food suppliers are described as 'competitors'.

The aspects of customer behaviour modelled in our system are specified in an XML. Customers are described in groups, allowing different types of customers to coexist in a single simulation. The output of the simulation that we are primarily interested in is the overall total numbers of orders received per day, though other factors such as the performance of customers in satisfying consumer demand and avoiding waste are also output.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>sell-by</td>
<td>How long before product must be disposed of</td>
</tr>
<tr>
<td>no-sunday</td>
<td>Does the company work on a Sunday</td>
</tr>
</tbody>
</table>

**Table 1: Overall Simulation Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>Number of customers in a group</td>
</tr>
<tr>
<td>average-demand</td>
<td>Average num. of units ordered/day</td>
</tr>
<tr>
<td>sd-demand</td>
<td>Standard deviation of demand</td>
</tr>
<tr>
<td>standard-order</td>
<td>Size of order of product in units</td>
</tr>
<tr>
<td>reorder-level</td>
<td>Level that triggers customer order</td>
</tr>
<tr>
<td>pActiveGivenInactive</td>
<td>Markov Model probabilities for loss and gain of customers, and initial customer base</td>
</tr>
<tr>
<td>pInactiveGivenActive</td>
<td></td>
</tr>
<tr>
<td>pActiveGivenStart</td>
<td></td>
</tr>
<tr>
<td>adaptive</td>
<td>Whether a customer uses demand prediction in ordering.</td>
</tr>
</tbody>
</table>

**Table 2: Customer Group Parameters**

Varying consumer demand (experienced by customers) is defined by 'daily delta' elements in the XML file. These define a 'delta' (e.g. 1.4) for each day of the week (e.g. Friday) The example delta of 1.4 indicates that on average a customer will consume 40% more product on a Friday than would be predicted by the average-demand and sd-demand parameters.

The status of customers is controlled by a Markov model. 'Live' customers are actively ordering from the company. 'Dead' customers are currently ordering from a competitor. Transition probabilities allow customers to change state.

**CUSTOMER DEMAND PREDICTION**

There are two types of demand in our simulation. 'Consumers' purchase products from 'Customers', and in turn 'Customers' purchase raw materials from 'The Company'.

Customer have their own demand prediction abilities, making them self-directed. Customers can abandon the simple strategy of re-ordering a fixed amount once supply of the product has failed to the re-order level. In the 'adaptive' mode, once the re-order level has been reached, the customer estimates their own demand over the next time period (e.g. one week or sell-by date) and orders from the company.
SIMULATION OUTPUT

The simulation creates a .csv file listing total orders received per day. If the company does not work on Sunday, Sunday orders are 0. Additional information includes the average and standard deviation of the orders, the SD as a proportion of average orders, and the error in prediction produced by a simple ARIMA prediction of daily demand as WAPE (Novobilski & Kamangar, 2001). ARIMA prediction is done on a day of the week basis, e.g. demand for a Tuesday is predicted from previous Tuesdays.

EXPERIMENTS AND RESULTS

The first experiment was to create a baseline accuracy for prediction based on average sales per day. A simulation run was run with the following parameters: steps=2500, sell-by=8, name="milk". One group of 200 customers was created with parameters: average-demand=1.0, sd-demand=0.4, standard-order=6, reorder-level=3, start-date=1st January 2009, active=false. Parameters were set so that all customers remained live at all times.

<table>
<thead>
<tr>
<th>Avg. Sales</th>
<th>Sales SD</th>
<th>Avg.Pred. Err</th>
<th>WAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>232.74</td>
<td>43.08</td>
<td>28.21</td>
<td>12.12%</td>
</tr>
</tbody>
</table>

Table 3: Experiment 1: Results of Simulation

The next experiment was identical except that the company did not work on Sunday.

<table>
<thead>
<tr>
<th>Avg. Sales</th>
<th>Sales SD</th>
<th>Avg.Pred. Err</th>
<th>WAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>230.74</td>
<td>57.69</td>
<td>23.70</td>
<td>10.27%</td>
</tr>
</tbody>
</table>

Table 4: Experiment 2: No Sunday Working

Average sales have decreased slightly, the standard deviation of sales has increased and accuracy has improved. This is because the 0 sales on Sundays are easy to predict.

In the above experiments, each of the 200 customers remains a customer of the company permanently, which is an unrealistically static situation. The average sales are over twice the average milk sales from the Zest data, and the WAPE error is lower than simple ARIMA applied to the real milk data. This suggested the following experiment which includes active/inactive customers. All parameters are the same as above, except for: pActiveGivenInactive=0.02, pInactiveGivenActive=0.02, pInitialActive=0.5 to enable the Markov model.

<table>
<thead>
<tr>
<th>Avg. Sales</th>
<th>Sales SD</th>
<th>Avg.Pred. Err</th>
<th>WAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>113.97</td>
<td>67.54</td>
<td>18.36</td>
<td>16.11%</td>
</tr>
</tbody>
</table>

Table 5: Experiment 3: Markov Model Enabled

The numbers of sales has dropped to less than half of that for the previous experiments. This was expected due to customers now being lost to competitors. With the probability of losing customers being the same as gaining customers, approximately 50% of customers will typically be inactive. Less expectedly the standard deviation of sales has increased dramatically, as has the error. This shows that the ebb and flow of the numbers of active customers has contributed significantly to the variance in ordering. The probability of losing a customer is low, with approximately 2% of customers being lost (but potentially gained) each day. However, the error (WAPE) has increased from 10.27% to 15.58%, an increase of 51.70% in the error rate.

The next experiment was to vary the daily demand that customers experience from consumers. The first of these experiments was to vary the daily order by a small amount. As there is only one group of customers, all customers experience the same variation in consumer demand. Table 7 shows the relative demand on different days of the week. Note that demand on Sunday is 0, simulating the customers being shut on that day.

<table>
<thead>
<tr>
<th>Day</th>
<th>Delta</th>
</tr>
</thead>
<tbody>
<tr>
<td>Monday</td>
<td>0.5</td>
</tr>
<tr>
<td>Tuesday</td>
<td>0.5</td>
</tr>
<tr>
<td>Wednesday</td>
<td>0.5</td>
</tr>
<tr>
<td>Thursday</td>
<td>2</td>
</tr>
<tr>
<td>Friday</td>
<td>2</td>
</tr>
<tr>
<td>Saturday</td>
<td>1.5</td>
</tr>
<tr>
<td>Sunday</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 7: Daily Deltas

Results are shown in Table 8.

<table>
<thead>
<tr>
<th>Avg. Sales</th>
<th>Sales SD</th>
<th>Avg.Pred. Err</th>
<th>WAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>113.93</td>
<td>58.22</td>
<td>18.05</td>
<td>15.84%</td>
</tr>
</tbody>
</table>

Table 8: Markov Model Daily Delta Typical

The above results are sensitive to the parameters of the reorder strategy. Previous experiments have been performed on a ‘like for like’ basis, but Table 9 shows how reorder levels and reorder amounts cause the overall prediction error to vary.

<table>
<thead>
<tr>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.85</td>
<td>15.93</td>
<td>17.33</td>
<td>18.54</td>
<td>19.58</td>
</tr>
<tr>
<td>13.18</td>
<td>15.49</td>
<td>16.59</td>
<td>17.54</td>
<td>19.80</td>
</tr>
<tr>
<td>12.84</td>
<td>14.70</td>
<td>16.36</td>
<td>17.06</td>
<td>18.50</td>
</tr>
<tr>
<td>12.74</td>
<td>14.47</td>
<td>15.52</td>
<td>15.71</td>
<td>17.54</td>
</tr>
<tr>
<td>12.32</td>
<td>13.56</td>
<td>14.74</td>
<td>16.24</td>
<td>16.69</td>
</tr>
<tr>
<td>12.19</td>
<td>13.53</td>
<td>17.25</td>
<td>14.44</td>
<td>15.37</td>
</tr>
</tbody>
</table>

Table 10: Prediction Error (WAPE) by Re-order Amount (columns) and Re-order Level (rows)
undertaken without the loss and gain of customers. The results are shown in Table 10.

<table>
<thead>
<tr>
<th>Avg. Sales</th>
<th>Sales SD</th>
<th>Avg.Pred. Err</th>
<th>WAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>245.46</td>
<td>122.46</td>
<td>24.00</td>
<td>9.76%</td>
</tr>
</tbody>
</table>

Table 10: Adaptive Customers Results

Another experiment was to repeat the adaptive customers, but to allow customers to be lost and gained by enabling the Markov Model. The results are shown in Table 10.

<table>
<thead>
<tr>
<th>Avg. Sales</th>
<th>Sales SD</th>
<th>Avg.Pred. Err</th>
<th>WAPE</th>
</tr>
</thead>
<tbody>
<tr>
<td>123.61</td>
<td>51.96</td>
<td>18.45</td>
<td>14.93%</td>
</tr>
</tbody>
</table>

Table 11: Adaptive Customers Results with Markov Model

The final experiment was to repeat the adaptive customers, but to allow customers to be lost and gained by enabling the Markov Model. The results are shown in Table 11.

CONCLUSIONS

It was possible to create a set of simulation parameters that matched the variability seen in real world ordering. While a matching of variability to the real world is encouraging. However it far from guarantees that the included in the simulation are correct, as the match to real-world behaviour could be fortuitous.

It was expected that introduction of coordinated demand for customers would lead to the company's demand being more predictable. That the prediction accuracy did not improve shows that the variance caused by customers going out of sync is not prevented by common influencing factors. This raises the question of whether other common influencing factors such as weather may not feed through to total customer orders received by the company. This is a potential cause of demand prediction being difficult and the lack of correlation observed between `obvious` factors such as weather and daily demand. Simply put, it appears that the effect of such factors is being lost in the noise created by the loss and gain of customers and in customers varying their day of order. It was expected that unequal consumer demand experienced by customers would tend to synchronise customers and improve predictability. That this didn't happen is an important, though discouraging result of this research.

The `adaptive` customers used a very different method for deciding how much product to order, which would have had effects on their reordering patterns. The small, but notable improvement in prediction accuracy cannot only be attributed to automatically choosing different reorder amounts, as typically error percentages do not vary that much across a range of reorder amount values. The adaptive strategy has the potential for a greater variance in the amount ordered. That this has improved predictability of demand is encouraging.

FUTURE WORK

The present simulation covers a single product, e.g. “milk”. In the real world customers will order a number of products from the same or varied suppliers. E.g. salad ingredients, fruit, prepared products such as chips, etc. There may be efficiency gains from ordering a batch of products at a single time for a single delivery. Cross-product effects may affect demand patterns, which may affect the demand prediction (done by the Company) for individual products. Investigating this further is important future research.

In future, as well as producing an `overall` order for a product through simulation, more investigation of individual customer behaviour is planned. See, e.g. (Clement et al 2015) for a description of clustering customers based upon ordering styles.

There is no shortage of additional simulation parameters that could be added to the system, exactly which parameters to add is a topic for careful future research.

ACKNOWLEDGEMENTS

The authors would like to thank InnovateUK (Project No: 101397 and Reference: 23792-161273) and De Montfort University for funding this research.

REFERENCES


LATE PAPERS
MODELING AND SIMULATION OF TAWAF AND SA’YEE: A SURVEY OF RECENT WORK IN THE FIELD

Omar Kurdi and Mike Stannett
Department of Computer Science
The University of Sheffield
Regent Court, 211 Portobello
Sheffield S1 4DP
United Kingdom
E-mail: acplloak@sheffield.ac.uk

Daniela M. Romano
Department of Computing
Edge Hill University
St Helens Rd, Ormskirk
Lancashire L39 4QP
United Kingdom
E-mail: Daniela.Romano@edgehill.ac.uk

KEYWORDS
Modeling, Simulation, Agent-Based Modelling, Crowd modelling, tawaf, sa’yee, hajj, umrah.

ABSTRACT

Between 2002 and 2012 the number of pilgrims taking part in the 5-day hajj (the annual pilgrimage to Mecca) rose dramatically from 1.9m to 3.2m, before stabilizing at around 2 million following the introduction of new quotas in 2013. The gathering together of so many people has obvious crowd-safety implications, ranging from stampedes and protests to pickpocketing and disease control, and there is an obvious need for models and simulations of the relevant crowd behaviours. Moreover, the regular occurrence of the event, the size and diversity of the crowds involved, and the amount of freely available information make this an excellent case study for the study of crowd behaviour. We survey recent attempts to model the key hajj rituals of tawaf (during which pilgrims collectively circumambulate the Ka’aba seven times) and sa’yee (running or walking seven times between two nearby hills), and highlight ways in which some of the limitations of these studies may be overcome in future work.

INTRODUCTION

One of the five pillars of Islam is the 5-day hajj, or annual pilgrimage to Mecca. Each adult Muslim is expected to perform the hajj at least once during their lifetime, unless prevented by health or financial considerations. To put this in perspective, in 2010 Muslims constituted over 23% of the global population, or more than 1.6 billion people, and this number is forecast to rise to around 2.2 billion by 2030 (Grin and Karim 2011). Clearly, therefore, the number of visitors to Mecca during the hajj is considerable, and can be expected to rise still further in the coming decades (pilgrimages at other times of year are referred to as umrah, and similar considerations apply). This has obvious implications for crowd control and safety, and many attempts have consequently been made to model and simulate the relevant crowd behaviours. In this paper we survey recent work in this area, and highlight ways in which some of the

limitations of these studies may be overcome in future work.

THE TAWAF AND SA’YEE RITUALS

The hajj, or annual pilgrimage to Mecca, involves the performance of several interlinked rituals, of which two of the most crowd-intensive are tawaf and sa’yee. Both rituals are performed on specific days each year at Islam’s most sacred mosque, the Masjid al-Haram, in Mecca. This is a large complex which currently covers some 356,800 square metres, and work in underway to expand this to 1.1m square metres by 2016, by which time the mosque will be capable of housing 2.5m worshippers (Cherupapa 2015).

During tawaf (“circling”), each pilgrim circumambulates the Ka’aba, a cuboid building at the centre of the mosque, seven times in an anticlockwise direction. The area around the Ka’aba where this is carried out is called the Mutaaf. This is followed by sa’yee (“ritual walking”), during which pilgrims run or walk seven times along a corridor joining the nearby hills of Safa and Marwah. The corridor and hills are incorporated within the mosque, in an area called the mas’ a gallery. The requirement that tawaf and sa’yee be performed as part of a coordinated series of rituals during the hajj leads to significant restrictions in both space and time, which in turn create high crowd densities throughout the tawaf ritual (Fig. 1).

THE CHALLENGE

There are many properties of the tawaf ritual that make its simulation difficult yet very interesting for the research field of crowd simulation, including complex flows of motion, changeable velocities, high densities and heterogeneous populations (Curtis et al. 2011). According to Janajrah and Virk (2014) the most significant factors include:

- the geometry of the Mutaaf area (Fig. 1);
- the location of key historical and ritual attractions;
- certain behaviours adopted by pilgrims, including chain-like movements and clustering,
and we may also include

- the size of the crowd gathered, containing in particular a high number of disabled individuals, as well as family and peer groups;
- the annual recurrence of the event, which allows us to make and test simulation hypotheses, and gather further evidence in the real world.

Figure 1: Three views of tawaf taking place within the Mataf region before (left) and after (right) the expansion project (Saudi Press Agency)

In particular, congestion at bottlenecks can cause pilgrims to move in conflicting directions simultaneously, giving a high risk of collision. This is further complicated by the heterogeneous nature of the population, since different groups of pilgrims move in different ways and by different means (quickly vs slowly; individuals vs groups; foot vs wheelchair). The development of appropriate models is essential to help identify triggers for key behaviors within dense crowds, e.g., trampling, falling, collisions, pushing, etc. In particular, the ability to simulate groups of people within the crowd is important, because such groups can act as obstacles in their own right.

Although we do not address it here, it is worth also noting the problem of validation – how do we know whether a simulation gives a reliable approximation to real-world behaviours? Current research relies on analysis of video footage, so as to identify real-world parameters against which simulations could be compared (Dridi 2014), e.g., paths followed by family members can be expected to cluster more than those of strangers. This task has been greatly simplified by the advent of modern technologies. For example, Koshak and Fouda (2008) used a variety of Geographic Information Systems (GIS) and Global Positioning Systems (GPS) to monitor the movement of pedestrians performing tawaf, and then used tracking-analysis software to analyze and visualize pilgrim movement patterns. As well as providing historical data for benchmark purposes, this kind of approach can be seen as a possible alternative to simulation, since it helps reveal real-world patterns that can inform future enhancements to the urban and architectural environment.

THE STUDIES

During the tawaf ritual, a large number of individuals move as members of an extensive crowd. It is natural, therefore, to break the system into two parts, one representing high-level crowd dynamics, and the other representing individual deviations from the norm. Curtis et al. (2011) proposed using a finite-state machine for high-level behavioural modelling, combined with an agent-level collision avoidance strategy. This approach allows the system to be calibrated by matching the overall distribution of simulated pedestrian velocities to observed values, while agents can be configured to exhibit behaviours like pausing to perform sub-routines, entering and leaving specific floors in the Masjid al-Haram, joining and leaving queues, and circling the Ka’aba. Their model behaves well even in the presence of excessive crowd densities, and many of the simulation outcomes matched those observed when watching tawaf performing by actual people. In addition, the system is general enough to allow a wide range of distinct behaviours to be introduced in future studies, such as agents representing different combinations of age and gender. Nonetheless, the approach has important limitations. For example, because the underlying focus in most crowd simulations is on individuals moving within the assembled crowd, it is not easy to discover what might happen if pilgrims gravitate together in small groups, as for example during tawaf. Such considerations are of paramount importance when modelling disaster evacuation scenarios, since group dynamics can seriously affect individual reasoning. Moreover, it is not enough simply to match overall velocity distributions, since these vary with the pilgrims’ location within the Mataf region, individual capabilities and group affiliations. Instead, we suggest that detailed velocity profiles need to be compiled for each of the main landmarks within the Masjid al-Haram, and for each type of pilgrim modeled, along with his or her group type. This leads to another approach to simulating crowd dynamics, which is to rely entirely on agent-based modelling (ABM). Rather than imposing high-level dynamics on the crowd, the focus is on modelling individuals and groups of individuals; running enough instances of crowd members then generates crowd behaviours as ‘emergent phenomena’. The basic geometry of the Masjid al-Haram in this case can be modelled by treating walls, barriers, attractions, etc., as agents in their own right, with specific ways of interacting with pilgrim-agents (Karmakar et al. 2010); other known navigation techniques can also be utilised. Khan and McLeod (2012) used this technique to investigate the effects of three parameters – the Masjid al-Haram courtyard layout, pilgrim crowd properties, and
the management preferences of the hajj authorities – on the safety (lack of collisions), health (spread of disease), satisfaction (calm environment) and throughput of pilgrims. This work provided many recommendations and insights for the crowd-management priorities during the hajj, and could be further by the development of an appropriate crowd simulator capable of modelling large-scale experiments, capturing the key behavioral metrics and features of the system.

Mulyana and Gunawan (2010) adopted a similar approach, simulating pilgrims as intelligent agents (Fig. 2), and again found that the simulation of the hajj crowd matched the practical performance of pilgrims during tawaf and sa‘ee. Their system has the additional advantage that it can be used to help train pilgrims before they travel to the Masjid al-Haram (other models can also be used for this purpose; see, e.g. (Yasin et al. 2012)). Although the additional computational requirements involved in simulating intelligence mean that pilgrim numbers currently have to be kept relatively low in the simulations, such problems could be alleviated in part by applying quantization within a virtual environment equipped with kinetic data structures and developing better collision-detection algorithms.

Figure 2: Intelligent Agents (Russell and Norvig 2009)

The activities of tawaf have also been modeled by Haghighati and Hassan (2013) using the discrete-event simulation tool ARENA. Pilgrims were represented as discrete units which repeatedly queue to enter and flow within the system, eventually leaving it after seven iterations (Fig. 3). Many factors were investigated within the model, including the behavior of switching throughout the tawaf, the availability of space, group sizes and inter-arrival times between pilgrims. Their results suggest that a key factor affecting crowd densification is the practice by pilgrims of lane-switching during performance of tawaf, and that this also reduced the efficiency of queueing strategies; this led them to suggest the use during tawaf of scheduling, spiral paths and clear separation. Indeed, their work suggests that organizing the tawaf to be performed along specific routes, and scheduling how pilgrims enter the Mataf region, will help prevent congestion and maintain crowd densities at sustainable levels, thereby generating a better experience for pilgrims.

Figure 3: Modelling the behaviour of individual pilgrims (Haghigaati and Hassan 2013)

Another approach giving realistic simulations is to employ concepts from virtual physics (Fig 4). Kim et al. (2015) treat pilgrims within the crowd as objects interacting with obstacles via physical forces, while enabling agents to anticipate collisions and take appropriate action to avoid them. Their approach reproduces several emergent properties and allows the simulation of a few thousand agents. While the agent numbers are small relative to the crowd sizes required for hajj simulations, the increasing efficiency of high performance computing resources offers hope that their work can soon be expanded to include systems involving millions of agents.

Figure 4: Simulation of tawaf by Kim et al. (2015)
Zainuddin and her colleagues have produced a series of useful studies in recent years. Zainuddin et al. (2009; 2010) used SimWalk, a software system that allows the design and analysis of flows through pedestrian areas, to model crowds operating under the Social Force Model (SFM), a complex approach based on the use of second-order differential equations to represent the forces acting on individuals. Their simulation allowed them to test, for example, a proposal (Al-Haboubi and Selim 1997) for the construction of a spiral path, showing that the use of such a path would allow circumambulation to be completed in around 21 minutes as opposed to 35 minutes without it. However, the study is limited by various built-in assumptions. For example, pilgrims are assumed always to follow the shortest possible path to complete their task, and the problem of crowd congestion at entrances is neglected. This latter problem is taken up in (Zainuddin and Aik 2012) which uses a Response Surface Methodology (Box and Wilson 1951) in which the Mataf courtyard is modelled as a grid of cellular automata which pass pilgrims from one grid location to the next according to current conditions. Their results suggest that the RSM approach is extremely promising, and although it is computationally expensive we believe it warrants further investigation.

Another study to investigate the effects of imposing spiralling motions on the crowd is that of Shuaibu et al. (2013), who considered the relative efficiency of circular motion during tawaf, as opposed to spiralling inwards for four laps along a specific spiral trajectory and outwards for the remaining three. Spiralling was once again found to be beneficial, since it reduced the simulated tawaf completion time from 60.1 minutes to 38.0, while simultaneously reducing crowd densities from 8.8 to 4.2 people per square metre. However, their work does not yet take account of certain key factors during the performance of tawaf, such as queueing and waiting at various traditional locations, and it also assumes that all motion occurs in a single 2-dimensional plane. As with other simulations discussed above, another significant factor is the relatively low number of agents modelled.

Although the approaches we have considered so far focus mainly on tawaf, Sakellariou et al. (2014) recently proposed ways to analyze, simulate and specify the performance of the crowd during sa‘yee. The approach adopted is to develop an abstract generic state-based model of crowd behaviour based on X-machines, a kind of extended finite state machine with applications ranging from the modelling of biosystems (Gheorghe et al. 2001) to the development of novel testing strategies for distributed hybrid systems (Stannett and Gheorghe 2015). Taking sa‘yee as exemplar, they show how models in their framework can be mapped into existing agent-based systems, in this case NetLogo (Fig. 5). The paper is also unusual in that it explicitly addresses the validation issue, and provides a rigorous approach for extracting data, parameters and behavioural patterns from video data. Moreover, the simulations described in the paper include as many as 70000 agents. It is worth noting, however, that the X-machine approach is particularly well adapted for use with the high-performance FLAME GPU agent-based modelling framework (Richmond and Romano 2011), and we envisage that adapting their approach to this framework would allow agent populations to be raised enough to make full-scale modelling of tawaf feasible.

`to-report state-def-of-persons` `report (list` `state "Entering"` `# r-func "follower?" goto "MoveToLeader"` `...` `end-state` `state "Walking"` `# r-func "reachExitSafe" goto "AtExit"` `# r-func "reachExitMaruh" goto "AtExit"` `# r-func "leaderFar" goto "MoveToLeader"` `# r-func "leaderFarTurns" goto "MoveToLeader"` `# r-func "followerFarMove" goto "WaitingForGroup"` `# r-func "followersFar" goto "WaitingForGroup"` `# r-func "moveToExit" goto "Walking"` `# otherwise do "nothing" goto "Walking"` `end-state` `(... more states ...)` `state "Exiting"` `# r-func "reachedEnd" goto "Exiting"` `# r-func "leavingCorridor" goto "Exiting"` `end-state` `)

Figure 5: Coding of an X-machine in NetLogo (Sakellariou et al. 2014)

Just as virtual physics has provided valuable insights for crowd simulation, so some researchers have turned to biology for inspiration. In contrast to other approaches based on individuals attempting to find the best route through a system, de Lima Bicho et al. (2012) simulate the competition for space between moving agents. Originally developed as a way to model the branching architecture of trees, the space colonization algorithm they use generates several realistic crowd behaviours, even though agents only observe free space rather than one another. These include (and hence potentially explain) several characteristic behaviours that are taken as starting points for other approaches, including collision avoidance, lane formation, and the relationship between velocity and crowd density. While this is certainly a very promising approach, the authors are careful to note some current limitations; as with other approaches, validation remains a problem, and the focus on individuals means that groups are neglected.

A problem we have repeatedly identified above is the inability of systems to handle enough agents to make life-size modelling of tawaf possible. Nonetheless, Narain et al. (2009) have demonstrated a technique that may overcome this problem in the long term. Their approach
is to represent the crowd simultaneously as both a collection of discrete agents, and a single continuous system to which a novel unilaterally incompressibility constraint is applied. This allows them to model large-scale behaviours with relative ease, making it possible to simulate crowd populations of several hundred thousand on a standard desk-top computer. While this is not quite large enough to model life-size tawaf populations, the increasing power of computer hardware again suggests that this will soon be possible. The authors also note a number of other limitations relating to the algorithms they use. For example, when two groups of agents approach each other, they will not react until their actually encounter one another – there is no in-built capability for groups to plan ahead and take early action against collisions.

![Figure 6: The dual-representation approach suggested by Narain et al. (2009)](image)

**DISCUSSION**

As we have seen, the need to model crowd behaviours during tawaf and sa’ee is pressing, and many techniques have been proposed. In general these focus on tawaf, since this involves large crowds of people entering a confined space via various uncoordinated entrances (Fig. 7), although a few studies have looked at sa’ee as part of a general analysis of crowd modelling strategies.

In each case the goal is to avoid collisions between pilgrims, to minimize the time it takes to complete each ritual, and to maximize crowd safety and well-being. The papers we have surveyed show a wide range of different ways to achieve these aims. While some researchers impose experimental data on the simulation, others seek to develop models in which the required properties ‘emerge’. Similarly, while some authors impose properties like collision avoidance on agent behaviour, others seek models that have collision avoidance itself as an emergent property. An important issue that is neglected in many of the studies is the existence of family and friendship groups within the crowd, and work is urgently needed to update the studies accordingly, since the existence of such groups can lead both to dynamic new obstacles during crowd evacuation situations, and to unexpected goal formation on the part of individual agents (e.g., parents may head towards a source of danger in search of children, rather than attempting to flee as would otherwise be the case). In this regard, it is essential that pilgrims be modeled not simply as automata, but as agents equipped with both intelligence and emotional states (Kefalas et al. 2012). Ultimately, however, all of these studies rely on the provision of accurate real-world data for validation purposes, together with the ability to run simulations that are large enough to make comparisons with reality meaningful. In this regard, we note that none of the models surveyed here successfully produce simulations with agent populations in the millions, as would be required for a life-size simulation of tawaf. Nonetheless, the increasing availability of cheap computational power, coupled with new modelling frameworks like FLAME GPU, offer hope that viable full-scale simulations will be within reach in the very near future. In the mean time, it is clear that further research is warranted, so as to combine the best features of each of the studies cited above.

**ACKNOWLEDGEMENTS**

This research is supported by the Royal Embassy of Saudi Arabia (London). We would also like to thank Professor Marian Gheorghe of Bradford University for his advice; his many insightful comments have proven invaluable during the construction of this paper.
### Table 1: Summary of Findings

<table>
<thead>
<tr>
<th>Study</th>
<th>Techniques</th>
<th>Model</th>
<th>Focus</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Namin et al. (2009)</td>
<td>CFD</td>
<td>Unilateral incompressibility</td>
<td>tawaf and other densely</td>
<td>Dual-representation, discrete agents overlaid on continuous high-level dynamics. Excellent performance, handles several hundred thousand agents easily.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Constraint; Lagrangian and</td>
<td>crowded areas</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Eulerian methods</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zainuddin et al. (2009; 2010)</td>
<td>ABS</td>
<td>Social-Force Model (SimWalk)</td>
<td>ta'awf</td>
<td>Uses SimWalk and the Social Force Model, shows that construction of a spiral pathway can significantly increase throughput for ta'awf. Neglects crowd congestion at entrances.</td>
</tr>
<tr>
<td>Muliyana and Gunawan (2010)</td>
<td>ABS</td>
<td>Agent based</td>
<td>ta'awf, sa'yec, jamarat</td>
<td>Models pilgrims as intelligent agents, and can be used to help train pilgrims prior to arrival. Computationally intensive, relatively few agents.</td>
</tr>
<tr>
<td>Curtis et al. (2011)</td>
<td>ABS</td>
<td>FSM, RVO</td>
<td>ta'awf</td>
<td>Finite state machine model of crowd dynamics, pilgrims modelled as agents. Behaves well with excessive densities. Simulates 7 key behaviors. Can be extended to include more complex agents.</td>
</tr>
<tr>
<td>de Lima Bicho et al. (2012)</td>
<td>ABS</td>
<td>FSM, RVO</td>
<td>ta'awf</td>
<td>Uses ideas from biology. Agents compete for space. Generates properties that are taken as starting points by other studies.</td>
</tr>
<tr>
<td>Khzn and McLeod (2012)</td>
<td>ABS, CA</td>
<td>micro-level Behavioral</td>
<td>ta'awf</td>
<td>Considers three parameters (courtyard layout, crowd properties, management preferences) and their effect on safety, health, satisfaction and throughput.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>algorithm</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Zainuddin and Alk (2012)</td>
<td>CA</td>
<td>Response Surface Methodology</td>
<td>ta'awf</td>
<td>Extends earlier work by including congestion. Based on Response Surface Methodology. Computationally expensive, but produces excellent results.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(RSM)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Shusibu et al. (2013)</td>
<td>Mixed Models</td>
<td>Social-Force model, RuleBased</td>
<td>ta'awf</td>
<td>Shows that spiral motion is more efficient than undirected circular motion during ta’awf, but assumes all motion occurs in a 2-dimensional plane.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>model, Cellular-Automata</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Haghhighati and Hessami (2013)</td>
<td>ABS</td>
<td>Discrete-event sys-tem</td>
<td>ta'awf</td>
<td>Shows the significance of lane-switching on crowd throughput.</td>
</tr>
<tr>
<td>Sakellariou et al. (2014)</td>
<td>ABS</td>
<td>X-Machines, Net-Logo</td>
<td>sa’yec</td>
<td>Develops a generic abstract approach and applies it to sa’yec. Uses X-machines and NetLogo. Could benefit from translation to FLAME GPU.</td>
</tr>
<tr>
<td>Kim et al. (2015)</td>
<td>ABS</td>
<td>Velocity Based And FSM</td>
<td>ta'awf</td>
<td>Uses ideas from physics. Allows simulation of a few thousand agents.</td>
</tr>
</tbody>
</table>

**Key:** ABS = Agent-Based System; CA = Cellular Automaton; CFD = Computational Fluid Dynamics; FSM = Finite State Machine; RVO = Reciprocal Velocity Obstacles
REFERENCES


PROCEDURAL MODELLING TECHNIQUES TO CONFIGURE SCENES
IN A SERIOUS GAME BASED DRIVING SIMULATOR

Pedro Rosa and Rosaldo J. F. Rossetti
Laboratório de Inteligência Artificial e Ciência de Computadores
Departamento de Engenharia Informática
Faculdade de Engenharia da Universidade do Porto
Rua Dr. Roberto Frias, s/n – 4200-465 Porto, Portugal
E-mail: {cil10109, rossetti}@fe.up.pt

KEYWORDS
Procedural Modelling, Driving Simulator, World Location, Serious Games, Road Safety.

ABSTRACT

There is a need to increase road safety to further prevent road accidents and casualties for the drivers and the pedestrians. In this paper we propose a solution to help increase road safety through serious games. We intend to create a framework capable of loading real-world location data and render it three-dimensionally in a virtual environment that can be used to conduct many kinds of studies. Such a 3D model will be then populated with virtual pedestrians and the user will be able to drive a virtual vehicle with Oculus Rift. Such a feature allows the user to interact with the environment and walkers. There is also a full day-night cycle and dynamic weather conditions in order to help simulate different visibility and adherence conditions, as well as a full dynamic damage system. Such features are expected to increase immersion of subjects when being evaluated in a serious-game-based driving simulation environment. The proposed approach can run on an affordable commodity personal computer as every tool used is open-source.

INTRODUCTION

Road safety is a very important problem needing further appreciation in what concerns prevention, especially when drivers are distracted and not paying attention to the road. Thus any help attempting to study human drivers and their behaviour on important aspects will greatly contribute to road safety. In order to tackle this problem we propose a framework capable of loading a real-world location in 3-dimension scenes that will be flexible enough to help engineers as well as practitioners to conduct different kinds of studies.

From this framework it is expected that it can import real-world data and render it in a virtual environment in 3 dimensions by means of procedural modelling techniques that will then be populated by pedestrians that walk around and vehicles traversing the streets of a network. When all the location generation is finished this framework should allow the user to drive a virtual vehicle and collect metrics while the user drives around. It is possible to configure such metrics on the main menu. Using these metrics, for instance the time the user is not paying attention to the road, represents a great contribution to study human drivers and their behaviour towards improving road safety somehow. Since the scene is going to be procedurally generated, it should be a good feature to allow the possibility to export the entire scene, so it can be used by an external 3D application. The main expected contribution of this work is to introduce procedural modelling techniques to speed up the process of setting up driving simulation scenes to be used in serious-game environments. Contrary to traditional driving simulation setups, which are generally very expensive to run and maintain, although much more limited in terms of experimentation capabilities, serious games have been introduced as an important resource to the study of subject players.

This work is organised as follows. Section II reviews the literature concerning important concepts to this study, such as driving simulators and their peripherals as well as procedural modelling. Section III presents the tool chain necessary to implement the proposed architecture, whereas Section IV details all its components. Section V illustrates implementation results, and Section VI draws conclusions and presents the next stages of this research.

LITERATURE REVIEW

Important concepts and tools are discussed in this section, which will underlie the proposed approach detailed later on.

Driving Simulators

A driving simulator is a very refined and complex system that recreates the driving experience of a real vehicle on a computerized environment. It uses a graphical library to represent the environment, whose structure is divided in a few models to simulate the behaviours entities. Some of these modules may act inside a general physics engine such as the tire module, which recreates the behaviour of a real tire, the suspension module, the engine and braking system modules. The real-time telemetry module gather all the information and displays it on a graphical way to show how the parameters are evolving (Mechanical Simulation Corporation n.d.).

There are at least three types of driving simulators. One is for training, another is for entertainment and the last one is for research. All can be used to analyse people’s behaviour and create statistics on certain conditions (Toyota Motor Corporation n.d.). Any kind of computer program needs an
input from the user to make something happen and driving simulators are no different. They need to have some sort of input peripheral so the user can control the simulation in an allowed and controlled way. These input peripherals play an extremely important role on the immersion of the user and on how he/she feels when using the simulator. The better the peripheral and its utility on a specific simulation the better will be the immersion of the user and the better will be the results of the simulation. Some of these peripherals also work as an output device that transmits useful information to the user. There is a huge variety of peripherals whose prices range from a few euros to thousands of euros. They also vary in terms of suitability, shapes, function and quality. Some examples of peripherals are Steering Wheels, Force Dynamic Simulators, Surround Sound Systems, Oculus Rift, and so on.

Procedural Modelling and Applications

Procedural modelling is the name for computer graphics techniques that create a 3D model or texture by means of a computational procedure that implements a computer graphics algorithm to generate that model. This is usually done in runtime or a priori, so as to generate a model that is stored for future use. The resulting model depends on the inputs of the procedure, on the sets of rules the algorithm obeys and on the goal of the model (Smelik et al. 2014). The biggest advantage of procedural modelling is the fact that it is a procedure that generates the content with no human intervention. Being so, it is possible for instance to generate an endless terrain that keeps being created as the user moves around (in the case of a game). Another advantage is that it saves storage space. Since the content is runtime generated in some cases, there is no need to store the generated content because it can still be regenerated by the same rule set. A disadvantage is that sometimes the content generation might not be perfect, in which case it would still need human intervention.

Terrain procedural modelling is one important use for this kind of techniques. Most of the applications that need a very big terrain also need to save disc space and development time, so to avoid occupying the entire disc or folder with terrain data. It is then more efficient to generate it in real time recurring to terrain generation algorithms rather than storing the entire terrain model (Smelik et al. 2014). A recent method (Peytavie et al., 2009) provides a more elaborate structure with different material layers that support rocks, arches, overhangs and caves. Their resulting terrain models are visually very plausible and natural (Smelik et al. 2014).

Procedural flora includes all the vegetation such as grass fields, trees, bushes and flowers. When generating a procedural terrain, the flora also needs to be generated. However, the terrain mesh generator cannot do it itself, unless it simply places pre-modelled vegetation. This will not give the best results though, as there will be no uniqueness among the objects, meaning that all will look like the same. Flora generation algorithms take care of covering a terrain, whether procedurally generated or not. These algorithms generate the vegetation meshes according to input parameters to get different results like pines and palms, or similarly such as groups of trees from the same species. Slight differences like the height, the leave density and so on, depending on the algorithm’s complexity, will be noticeable though.

Procedural buildings are also an important aspect of procedural generation, because most of the scenes that are based on cities need to have buildings. Their generation can be performed by joining and combining primitives, making a structure with constant width or larger at the bottom and thicker at the top (Kelly and McCabe 2006). Another method are the L-Systems, which assume that a building is generated by successive clips to its bounding box. The clipped sections of its box are each iteration smaller, meaning this is an excellent algorithm to create skyscrapers (Martin 2005).

Procedural road generation consists in generating a smooth mesh that adapts to the ground elevation. The road mesh can also contain sidewalks and be merged to other road meshes. There are two ways to generate procedural roads, the first is the grid layout, where roads are generated in a pattern of a uniform grid; this method creates a New-York-style road network. Greuter enhanced this method to create a less regular grid, with road angles different from 90° degrees as well as bigger blocks between each other (Kelly and McCabe 2006). The second method is as well based on L-Systems that consider the road a branching structure.

And lastly, procedural cities may be a combination of many or all of the procedural techniques referred in this paper. Indeed a city is a very complex geometry that needs different kinds of algorithms to generate the different components.

Procedural Modelling Techniques

An L-System, also called Lindenmayer System, was developed as a mathematical theory of plant development that consists of a finite set of characters belonging to an alphabet arranged in arbitrary length sequences to form strings. Each character is associated with a rewriting rule that is defined by a grammar. Now the string needs to be converted to a three dimensional model using a technique called turtle interpretation, which is based on an imaginary turtle that moves, turns and draws according to instructions given. The turtle has a well-known and defined 3D position and a heading vector that points towards the direction of the movement. Each character of the string is interpreted as a command by the turtle. There is a variant of the L-Systems called bracketed L-Systems that provide two extra characters that are usually square brackets (“[“,”]”) that are able to push and pop the turtle’s direction, position and other information in order to allow the generation of branching structures (Green 1993).

There are a lot more details behind the L-Systems, but this was just a little review on them introducing the basis upon which they work. There are very good articles that have much more detail such as Prusinkiewicz’s graphical modelling using L-Systems (Prusinkiewicz 1990).

Another technique is the fractals, which are a natural phenomenon that can be represented by a mathematical function that generates a repeating pattern. This pattern can be displayed at every scale. So no matter the scale, the pattern will always be present, making the fractal scale invariant. When the replication is the same no matter the scale the fractal is called a self-similar pattern. It generates the same pattern at smaller and smaller scales as the
recursion progresses. Another type of fractals is the Statistical fractals where the pattern repeats stochastically, so numerical or statistical data are preserved across scales. The latter can add a little randomness to the fractal meaning that it is not mandatory that the fractal pattern strictly repeats. This is useful to generate floras. These fractals can generate a tree by adding the log and fractioning it at each iteration until the leaves are reached. Now that we know what fractals are, there must be a way to generate them. In this case L-Systems can be used as they are fractal generators.

TOOLCHAIN

This section presents all tools and their role within the conceptualized architecture, as detailed later on.

Unity 5

Unity3D is the game engine chosen mainly because it has been used in other projects within the team and members are very familiar with its peculiarities and potentials. It is also very intuitive and simple allowing the user to do very complex tasks with just a little effort. There is also a huge amount of very good documentation, tutorials and even specialized forums, as it is a popular game engine with a considerable community sharing experiences and knowledge. Unity is an entire ecosystem of tools and services designed for people who want to build a successful business by creating multiplatform games and interactive content. Another advantage is its easy integration with Microsoft Visual Studio that allows for code to be written in a much more sophisticated IDE than MonoDevelop.

OpenStreetMap

OpenStreetMap is a map that is a trusty representation of the whole world built and enhanced by the community on a collaborative basis (Neis, Zielstra, and Zipf 2011). Being so, it is almost an open-source project with the main goal of sharing with everyone real-world map data so people can use it for whatever purpose they may want.

Google Static Maps API

Google Static Maps API allows users to download satellite images based on the parameters of a URL sent by HTTP. These parameters can be the geographic coordinates, the zooming factor, the address of the location and pretty much anything that Google Maps uses to refer to its locations.

CGIAR-CSI

CGIAR-CSI (http://srtm.cgiar.org/) is a free SRTM digital elevation data provider where all data has been processed to fill data voids, and to facilitate its ease of use. The free data provided is from SRTM 90m DEMs that has a resolution of 90m at the equator, and are downloadable in seamless mosaiced 5° x 5° tiles. All this data is available in both ArcInfo ASCII and GeoTiff formats to facilitate their ease of use in a variety of image processing and GIS applications and cover over 80% of the entire globe.

UnitySlippyMap

UnitySlippyMap (Derrough, 2012) is an open source project that aims at helping developers to create maps working with a variety of online tile providers including OpenStreetMap. It can create tiles on Unity’s 3D space along the XZ plane and has a flying camera. The map can be zoomed and dragged using the mouse and new tiles appear or disappear as needed. In order to include maps from OpenStreetMap in our application, UnitySlippyMap needs a few changes in the code. One of them was disabling all unnecessary features to produce a simpler- and easier-to-use interface. The second was to allow placing a marker on the map using the mouse; when the marker is placed, a second marker will define a diagonal that will be used to create a bounding box. This bounding box limits the area to be downloaded.

Blender

Blender (Ahmed and Janghel 2015) is a free modelling software and is used to build vehicle models. Any kind of enhancement to the vehicle’s mesh needs to be done manually as well as creating collision meshes that are much simpler models of the original vehicle.

ARCHITECTURE

The system architecture is depicted in Figure 1, showing all its components and their interactions. Each element of the conceived architecture is detailed in the subsections that follow.

![Figure 1: System Architecture Diagram](image)

Main Menu

The main menu provides the initial interaction between the user and the application and is extremely important, because it allows the user to choose options and modify parameters that will affect the generated scene. The main screen presents the user very fast options and buttons to start and quit the application as well as going to the advanced options and choose whether to height maps or not. The middle-left box provides an input box where the user can enter the location by address or by coordinates in degrees. The application only starts if the input box has a valid location. Some of the advanced options are the ability to toggle which objects are going to be rendered, the day-night cycle duration and starting time, the weather conditions and its change
probability, as well as the vehicle damage varying among none, visible-only and mechanical.

**Location Data Parsing**

A location file has the .osm extension, but it is nothing more than a regular XML file. Being there are tools capable of parsing such files almost automatically. C# contains a library belonging to System.Xml that allows creating an instance of an XmlTextReader, which opens a file and starts reading it node by node. The file contains multiple different nodes, but only a few are relevant for this project. The node “bounds” refers to the bounding box of the downloaded area; its attributes are the minimum and maximum latitude and longitude. The node “node” refers to a node of the map. Its relevant attributes are its location (latitude and longitude) and its ID. It can have multiple sub-nodes called “tag”. A “way” node represents a group of “node”. It contains sub-nodes “nd” that represent a reference to “node” nodes. A “way” can also have various “tag”. A “tag” identifies what the “node” or the “way” are. The other nodes are not relevant. The parser takes into account all this information and is capable of clustering tags in order to correctly identify what a way and a node are.

**Satellite Image Gathering**

Google Static Maps provide various images for the same location with different zoom levels. The zoom level used was 18, which is the maximum for the free version of the API and is capable of covering a 382x382 meter tile. Having this zoom level means the images only cover a small part of the location and in order to download a consistent detailed photograph of the entire location the images have to be downloaded by tiles. A tile is a square plane, which has a satellite image as a texture. Knowing the bounding box limits and that the satellite images cover a constant distance of 382 meters we can iterate on 382 meters, starting from the minimum latitude and longitude until we reach the maximum latitude and longitude. For each 382 meters a tile is created and positioned on the respective distance from the centre of the locations bounding box and a request is made to Google Static Maps to download the image. As soon as the download of the image finishes, a material is created using this image as a texture and this material is then applied to the tile.

**Terrain Elevation**

This module uses the data gathered from CGIAR-CSI to create tiles that have the correct elevation for the specified location. These tiles are a bit different from the tiles described on C, because they are not a single plane, but instead they are groups of procedural-generated polygons. For each tile, the application runs an algorithm that uses the elevation ASCII file and creates a window to filter the needed values corresponding to the values between the minimum and maximum coordinates of the location’s bounding box. Each tile is composed by 16 polygons. Let D be the distance equal to 382 meter and EDS the elevation data spacing equal to 90 meters. Then, the number of polygons to cover one direction is D/EDS, which is equal to 4 polygons. So according to the calculation above we can easily see that a tile is composed by 4x4 polygons. Each tile can be considered a grid where the central point is the longitude and latitude of the entire tile. Knowing that the cell size of the elevation data is always 90 meters we can calculate the position of all vertices around that center and get the respective height from the tile. When all the information is ready the algorithm starts generating polygons for the given groups of four vertices. These polygons do not need to be translated into the correct position because the vertices are already on the correct place, so the polygon is generated accordingly.

**Procedural Roads**

When all the roads are identified as so and stored in a list, a function reads all the elements of this list and for each one a procedure is called to generate it. A problem is that nothing about its type and look is known, so this procedure parses all the tags of that path and attempts to figure out the road properties. There are a few techniques to generate a road mesh, such as Bezier curves, round splines and even grid layout. This is all very good for completely procedural scenes, but this project is not the case, because we already have all the road nodes.

Having these nodes restricts the road’s shape and roundness. With Bezier curves we could not have hard bends or square corners and the round splines imply having something round, which we do not want. The solution found was actually very simple.

To generate the road mesh the procedure creates control points along the path and two more points that follow along with a distance equal to the road width. When the entire path is processed the next step is to smooth the path to make it look more realistic. This method attempts to smooth a road by making an average of its points with some rules. The first set of points to be smoothened is the control points, because they represent the middle of the road. The procedure picks the first and the third control points and calculates their average position by adding the positions of the two points and dividing them by two. The second control point will be equal to this average. Then the procedure goes on to the next point and the process repeats.

For the road vertices, the average is calculated the same way. The only difference is that the point needs to maintain its distance from the control points (half the width of the road). When the smoothing is finished, it is time to pass all the vertices to a procedure that will use them to generate a mesh. The sidewalks are generated exactly the same way as the roads; the only difference is that they have an offset equal to the road width and they use the points of the road in order to follow along with no void gaps in the middle.

**Road Enlightenment**

The illumination is generated by two types of street lights, regular street lamps and the ones from the city. Each of these lights cast shadows and is displayed with different levels of detail depending on the distance to the main camera. The lights are placed close to the road, usually on the sidewalk gap on a regular interval on both sides of the street. They are
generated as soon as the road they are going to be placed on finishes rendering. When all the roads are rendered, there is an algorithm that gets all the street lights and for each one it casts a ray down from the position of the light and if this ray crosses more than one road, then probably this is a zone where two or more roads overlap, which means that the street light needs to be removed from the scene. This prevents having a street light in the middle of an intersection for instance. This project also features a full day-night cycle, meaning lights turn on and off when needed according to the time of the day.

**Pedestrians**

This project runs a simple pedestrian system that attempts to simulate small crowds of different persons with different behaviours. After a road is generated, a script is attached to every walkable element, in this case sidewalks and footways, with their respective vertices. Therefore these vertices are easily accessible as we know the game object they belong to.

Then an algorithm creates the pedestrians based on the pedestrian density configured on the main menu, by picking a random walkable area and a random vertex belonging to that area to define the position where the pedestrian is created as well as a random human character. Pedestrians have an AI script attached that controls their possible states, movement, object avoidance and route changes. The walker element can be in one of four possible states that identify if it is idle, walking, running or has been hit by the vehicle. If the pedestrian does not have a path defined yet, then it will calculate its own path over a walkable surface.

A pedestrian always has a walkable game object associated with it, but that does not mean its path along that walkable surface is calculated. When the pedestrian is over more than one walkable area it has the freedom to choose whether to change to another walkable area or to keep on the same one. The walker also runs to a walkable area if it knows to be walking on a road. When they get hit by the vehicle they turn into a ragdoll. This changes their state to the “hit-by-car” state and counts a casualty for the statistics. The ragdolls have a massive impact on physics calculation; in order to enhance the frame rate to a certain level of detail (LOD), a controller was created to enable or disable the walkers according to their distance to the camera.

**Procedural Buildings**

Every procedural building has the basics of any architectural structure such as walls, floors, basement and windows. The basement is responsible for aligning the building with a non-planar surface. The building height is usually more or less proportional to the bottom area. In average it can be up to 4 times the bottom area divided by the height of the floors. In this case, since the area is a complicated calculation, because of the shape of the path, the algorithm uses the perimeter of the path and a few random factors to determine the height.

In this project a building is considered a skyscraper if it has more than fifteen floors. The perimeter of the path is a key feature in what concerns defining the appearance of a building. This value is used as a seed of a random number generator; for the same building/perimeter ratio, the final result will be unique.

The materials of a skyscraper are different from a regular building. The former has a texture with lots of windows, whereas the latter only has texture for the walls, so skyscrapers do not have window meshes, contrarily to the regular. By generating a window mesh instead of using a texture it is possible to configure its size and position along the walls of the buildings. Windows are generated along all walls and floors and in the creation process they are equally distributed along each wall avoiding overlapping any doors.

**Procedural Vegetation**

In this project the main concern is not the appearance, but the functionality, so the vegetation is predefined trees and bushes that are generated in areas marked as forests. Such areas are closed paths and their area values are the seed of the random number generator, so vegetation will look randomly distributed; however, it will have the same result for the same area at every execution of the algorithm. Nonetheless, vegetation also adapts to the terrain elevation.

**Procedural Urban Furniture**

Urban furniture is a very big field requiring many different 3D models, so to simplify, in this project only garbage and recycle bins are generated. They are predefined objects that are physically simulated and placed at identified nodes of the map.

**Day-Night Cycle**

This project has a full day-night cycle that is a necessary feature to perform studies under different lighting conditions. On the main menu, it is possible to configure the duration of the day in minutes and the starting time. This system also integrates an event handling framework that triggers events to turn on and off the street illumination when the sun sets or rises.

**Weather Cycle**

This project features a dynamic weather system capable of changing conditions over time, interpolating between them. Weather states can be chosen in the main menu, including sunny, cloudy, foggy, rainy, stormy, snowy and snowstorm as well as the weather change probability. The duration of the weather states and the velocity of the transitions depend on the duration of the day. The shorter is the day, the faster are transitions and changing rates. Fog can be present in any other state and its intensity is a random value. All the different conditions have direct impact on handling the vehicles.

**Vehicles, Illumination, Cameras and Damage**

In order to explore different vehicle drivetrains and respective characteristics, a few vehicles are included, such as cars, trucks, 4x4 and buses. Each type is featured with a full set of lights, respecting international traffic rules for legal illumination of the road and a set of cameras, ranging
from cockpit vision (with Oculus Rift integration) to bird-eye view. Also, all vehicles have damage modelling, both visual and mechanical, which can be switched on and off. Mechanical damage can cause misaligned direction, flying wheels, power loss and broken suspension, for instance.

Oculus Rift Integration

The internal camera of all vehicles features Oculus Rift integration in order to create an extremely immersive experience for the users. Such integration therefore produces much better results. Users nonetheless can switch this feature on and off, on the main menu.

Scene Export

The entire scene is procedurally generated, so as to facilitate its further use or manual enhancement. This project allows the possibility of exporting the entire scene as “.OBJ” file that can be opened and edited by 3D modelling applications, such as Blender.

IMPLEMENTATION RESULTS

The proposed architecture has been effectively implemented and is fully functional. Scenes resulting from the procedural modelling approach, as described in this paper, are illustrated in figures below. Even though artificially generated in a game environment, scene graphics are rich enough to promote considerable immersion to the player, who will be able to identify main features. Such a degree of resemblance will certainly impact decisional processes at the game level.

In images shown in Figures 2 to 7, it is possible to notice that the application looks very appetitive, but images do not show everything in detail. In fact, the project is very functional, fulfilling all requirements as initially defined.

The day-night cycle works smoothly, and the weather conditions change very well between different states. The dynamic environment may affect vehicle’s performance directly, as weather conditions can change from clean sky to rain; the damage system is also very dynamic and flexible. Buildings look good with all floors and windows placed on the correct positions, even at a distance; they could certainly look better using higher resolution textures, for instance. The vehicle handling mechanism is also open for further enhancements. The Oculus Rift integration is sufficiently immersive that makes the user feel inside the vehicle, so the driver analysis is much more reliable. The rendered scenes are very pleasing, with everything working in harmony to produce an appropriate driving simulator based on a game engine, such as Unity3D. But of course, there is always room for further improvements.
CONCLUSIONS AND FUTURE WORK

There are a few attempts to do a 3D reconstruction of real-world data using OpenStreetMap, but none of them does it so deeply and they are far away from having any kind of population of vehicles or walkers. Goetz and Zipf transform the OpenStreetMap buildings into polygons with no windows and no texture, and there are no roads and no vegetation (Goetz and Zipf 2012). Two more examples are (Goetz and Zipf 2012) and the work by Neubauer, et al. (2009). Comparing our approach to these examples, this project advances a step further as it combines procedural modelling of real-world data and driver behaviour analysis. It also offers the possibility of generating a scene that can be exported as a 3D object file, which can be opened by 3D modelling applications, such as game engines.

The rapid prototyping of experimentation setups to test with some decisional aspects of players can have a great impact and help a better understanding of road safety strategies (Gonçalves et al. 2012; Alves et al. 2013). This approach does not intend to replace traditional driving simulators. It is rather presented as a complementary basis to support the elicitation, assimilation, and persuasion of subject players’ behaviour (Rossetti et al. 2013; Rossetti et al. 2011). Therefore, the very next stages in this research consider the integration of this approach with traditional (real-scale) driving simulators as well as with microscopic mobility simulation models, within the MAS-Ter Lab Framework (Rossetti and Bampi 1999; Rossetti et al. 2007). As scenes and networks are aligned in these three different perspectives, more complex experimentations towards addressing various issues in road safety analysis can be carried out, as well as autonomous driving simulation (Pereira and Rossetti, 2012).

REFERENCES


WEB REFERENCES


AN EXTENSION TO THE FEEDB MULTIMODAL DATABASE OF FACIAL EXPRESSIONS AND EMOTIONS

Luis Marco-Giménez, Miguel Arevalillo-Herráez  
Departament d’Informàtica  
Universitat de València  
Avda. de la Universidad s/n  
46100 Burjassot, Spain  
Email: margi4@alumni.uv.es, miguele.arevalillo@uv.es

Aladdin Ayesh  
Faculty of Technology  
De Montfort University  
Leicester, LE1 9BH, UK  
Email: aayesh@dmu.ac.uk

Mariusz Szwoch  
Gdańsk University of Technology  
Department of Intelligent Interactive Systems  
Poland  
Email: szwoch@eti.pg.gda.pl

KEYWORDS
FEEDB, Emotions Database, Kinect

ABSTRACT
FEEDB is a multimodal database that contains recordings of people expressing different emotions, captured by using a Microsoft Kinect sensor. Data were originally provided in the device’s proprietary format (XED), requiring both the Microsoft Kinect Studio application and a Kinect sensor attached to the system to use the files. In this paper, we present an extension of the database. For a selection of recordings, we also provide a frame by frame analysis in text format, that allows one to use the data directly for classification purposes. The data provided includes many different features computed from the original recordings, namely tracking status, 6 animation units, head position, head pose (pitch, roll and yaw) and 100 tracked points. These were extracted by using the Microsoft Face Tracking capabilities and are provided for each frame.

INTRODUCTION
Affective computing is a very active research area focusing on producing applications that take in consideration the user’s affect states as part of the interaction model. Methods to recognise affect may use different sources of information e.g. video [1], [2], audio [3], standard input devices [4], [5], [6], [7], physiological signals [8], [9] or depth information [10]; which may also be combined to obtain more reliable results [11], [1].

The major motivation to create a new version, was the format of the recordings. The first version used the proprietary XED format whilst the second version is using the accessible AVI format. Although AVI is accessible format, the processing of video frames for the purpose of image understanding is far more difficult not least for the lack of reliable supporting tools, e.g. Facial recognition libraries.

In contrast, XED files can easily be processed by using Microsoft Kinect Studio (MKS) and the Kinect for Windows Software Development Kit (SDK). MKS can be used to replace the Kinect’s input by the one recorded in a XED file; and the SDK provides useful face tracking operations that can be used to extract meaningful data from the recordings. However, the SDK requires a Windows PC with a Kinect device attached to it (serving as a licence).

To overcome this difficulty, we have used a specific setup
to process XED files and extract relevant data which can be used for emotion classification. This information is provided in text files and expands the possibilities offered by the first version of the database. The reason for following this route is motivated in the next section. The paper then covers the process of data extraction, the limitations of the tools developed, and then discuss the results of running these tools over the FEEDB database.

**Motivation**

In [2] we have used Kinect SDK capability of producing limited number of AU’s to translate video recordings to a data set of AU’s over which data classification algorithms can be applied. There are two benefits gained by doing so: first, we can process data in complete separation from the platform; second, we can utilise the wide range of existing classification techniques.

Applying wide range of classification techniques with no or little modification means we can focus on the interrogation of data and extraction of knowledge. This also minimize the amount of energy spent on getting algorithms to work with strict formats that has limited accessibility. Producing a stand alone standard data will also enable accessibility and re-usability for other data mining objectives in other projects and by other researchers.

As the work progressed extended database was required preferably in open data format. Another problem we faced was the limited number of AU’s provided by the standard SDK whilst it provides rich data of mesh points. It became clear that we needed a means of capturing and pooling this rich data.

**Data Extraction**

Microsoft distributes a free Face Tracking SDK that supports the creation of software that can track human faces in real time. This SDK is able to track one hundred 2D points, some of which are illustrated in Figure 1. This SDK is hardware bound to Kinect and can only be used with the presence of a working Kinect system.

The SDK reports the position of the user’s head, using a right-handed coordinate system with the origin at the sensor, the z-axis pointing towards the user and the y-axis pointing up. In addition 3D head pose angles (pitch, roll and yaw) are provided; along with the weights of six Animation Units (AU) and eleven Shape Units (SU), which are part of the Candide 3 model [14].

We have used the face tracking capabilities of the SDK to process each XED video and extract additional information that can directly be easily processed in text format. This has been done in two steps, which are illustrated in Figure 2. First, we used the Microsoft Face Tracking SDK to create a real time command line tool in Visual C++ language. The main purpose of this application was to eliminate the need to have a Kinect device connected to the PC to be able to work with the recordings. The application was able to read the Kinect’s input and generate a binary file in real time. This binary file contained the color and depth frames, along with all the face tracking information described above.

Fig. 1: Some of the 2D Points Tracked by the Microsoft Face Tracking SDK

To process the data in the XED files, the Microsoft Kinect Studio application was used to play their content, at the same time as the command line tool was started. The reason for using binary format was to reduce both the application’s throughput and the processing requirements. Still, this was a critical operation, and required the use of a Solid State Drive (SSD) to run at 30 frames per second.

Resulting files were then processed by using a second graphical application that was able to read the binary files produced by the command line tool and create appropriate data structures to hold their content. These were then used to support data visualization and format conversion (see Figure 3), and did not require a Kinect device attached to the machine. In particular, this graphical tool provided exporting facilities that made it possible to extract the following data for each video frame (all in text format):

- **ISTRACKING.** Reports whether the face tracking is active (1) or not (0). If the face tracking is not active, the rest of the data is invalid.
- **TRANSLATION.** Indicates the position of the user’s head in the camera space (see Figure 4). Units are in meters, and the reference point is the sensor.
- **ROTATION.** Refers to the pitch, roll and yaw (rotations along the X, Z and Y axis, respectively). They are given in degrees and in this order.
- **BOX.** The X and Y coordinates for the upper-left corner and lower-right corner of a bounding box containing the face, in pixels and the video space (see Figure 4).

1 More information about the meaning of these can be found at the Face Tracking SDK documentation (http://msdn.microsoft.com/en-us/library/jj130970.aspx).
Fig. 2: Two Step Data Conversion Process

Fig. 3: A Screen Shot of the Application that allows Data Visualization and Format Conversion
• FEATURES. The X and Y coordinates for each of the 100 tracked points, also in the video space.
• AU. 6 floating point values that correspond to the activation level of the 6 animations units provided by the Face Tracking SDK.
• SU. 11 floating point values that correspond to an estimate of the particular shape of the user’s head: head height, eyebrows vertical position, eyes vertical position, eyes width, eyes height, eye separation distance, nose vertical position, mouth vertical position, mouth width, eyes vertical difference and chin width.

![Diagram of camera and video space](image)

Fig. 4: (a) Camera Space (values in meters) vs (b) Video Space (values in pixels, in 640 x 480 resolution)

All the files produced follow the same structure. Each row represents a frame, and data items within the same frame are separated by spaces. This allows on to easily load the files in Matlab, Excel or other common analysis application. In Excel, files can be loaded using the import from text option, and selecting the space as the separation character. In the specific case of Matlab, files can be loaded by using the load command, with the name of the files in brackets. Figure 5 shows an example of a plot generated in Matlab.

![Example of plot](image)

Fig. 5: A Matlab Example

The number of data items (per frame) in each file are summarized in Table II.

<table>
<thead>
<tr>
<th>File Suffix</th>
<th>Data Items Per Frame</th>
</tr>
</thead>
<tbody>
<tr>
<td>ISTRACKING</td>
<td>1</td>
</tr>
<tr>
<td>TRANSLATION</td>
<td>6</td>
</tr>
<tr>
<td>ROTATION</td>
<td>3</td>
</tr>
<tr>
<td>BOX</td>
<td>4</td>
</tr>
<tr>
<td>FEATURES</td>
<td>200</td>
</tr>
<tr>
<td>AU</td>
<td>6</td>
</tr>
<tr>
<td>SU</td>
<td>11</td>
</tr>
</tbody>
</table>

**Limitations**

The Kinect device is able to work in two modes. In default mode, it offers best accuracy for objects in between 80 cm at 4 meters from the camera. By using near mode, best accuracy is achieved for objects in a range from 40 cm to 3 meters. As the tracking operation relies on the information captured by the sensor, tracking operations do not work for subjects closer than 40 cm. In addition, face tracking works when the user’s head pitch, roll and yaw are below 20, 90 and 45 degrees, respectively; and works best when these are less than 10, 45 and 30 degrees, respectively. The Face Tracking SDK also calculates a set of eleven values, which are used to adapt a standard 3D model so that it matches the shape of the face detected. These are called shape units(SU), and are part of the Candide-3 model [14]. A SU is related to a particular aspect of a human face, such as the eyes width and height, the vertical position of the nose and mouth, or the distance between the eyes. Learning the shape units for a given user in real-time takes approximately 2 minutes.

In FEEDB, most recordings were captured with users looking at the camera. This implies small values of pitch, roll and yaw, generally within the recommended range. However, many samples were taken at a distance close to 40 cm. In this cases, the face tracker is not able to track the user’s head, or constantly switches between on and off and does not produce usable data. For this reason, we have only been able to extract useful data from 11 of the 50 individuals who participated in the first version of the database.

Despite that FEEDB contains videos of 33 facial expressions and emotions, we have focused our work on 17 expressions that we have considered to be relevant to learning systems, which is the chosen user-end application for testing our affect and emotion classification algorithms [6], [7], [15], [2]. These expressions are: surprise (neutral), surprise (moderately positive), surprise (negative), smile (positive), smile (weak positive), smile (negative), pleasure, excitement, sadness, yawn, doubt, boredom, frustration, anger, concentration and attention. Recordings for other emotions such as speech, cry, fear, alight or irony have not been analysed.

We have been able to process most of the emotions considered for 5 participants (10 or more). In the rest of the cases we have only been able to extract data for a relatively small
TABLE III: Recordings with Additional Data

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>490</td>
<td>x</td>
<td>x x x x</td>
<td>x</td>
<td>x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
</tr>
<tr>
<td>378</td>
<td></td>
<td>x x x x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>518</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>669</td>
<td>x</td>
<td>x x x x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>388</td>
<td></td>
<td>x x x x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>363</td>
<td></td>
<td>- - - - x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>478</td>
<td></td>
<td>- - - - - x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>317</td>
<td></td>
<td>- - - - - x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>290</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>374</td>
<td></td>
<td>- x x x x x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
<tr>
<td>473</td>
<td></td>
<td>- - - - - x</td>
<td>x x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x x</td>
<td>x x x</td>
</tr>
</tbody>
</table>

samples: 4 5 5 5 5 7 2 3 4 6 6 6 7 4 7 7 6 5 88

subset of the 17 facial expressions, ranging from 1 to 7. In all cases, the length of the recordings was not sufficient for the system to compute the SUs and hence these values are not provided.

RESULTS

Our analysis has produced additional data for a total of 88 subject-emotion combinations, which are detailed in Table III. This data have been included in an extended version of FEEDB, with the aim of providing valuable additional information that can be used to clarify the fragile and elusive nature of affect.

Despite that the fine granularity of the original database has yielded a relatively low number of samples per emotion, a diversity of strategies may be used to group labels in Table III so that the can be used in a typical classification set-up. Table IV describes the number of samples and subjects per category when emotions are grouped according to a simple scheme based on their similarity.

TABLE IV: Emotions and States Grouped by Categories

<table>
<thead>
<tr>
<th>Emotion / State</th>
<th>Data samples</th>
<th>Subjects</th>
</tr>
</thead>
<tbody>
<tr>
<td>Surprise (01+01b+01c+01d)</td>
<td>19</td>
<td>7</td>
</tr>
<tr>
<td>Happiness (02+02a+02c+04)</td>
<td>16</td>
<td>8</td>
</tr>
<tr>
<td>Excitement (05)</td>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>Sadness (07)</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>Boredom (13+15)</td>
<td>10</td>
<td>6</td>
</tr>
<tr>
<td>Frustration (14+17)</td>
<td>14</td>
<td>8</td>
</tr>
<tr>
<td>Anger (18)</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Concentration (19+21)</td>
<td>11</td>
<td>8</td>
</tr>
</tbody>
</table>

Another widely accepted framework consists of considering the valence and arousal for each emotion [16], and use class labels that refer to intervals in each dimension. This approach has been used in e.g. [17]. In this way, the 8 labels in Table IV can be mapped into three classes in each dimension, as shown in Tables V and VI.

TABLE V: Emotions and States Mapped in Classes on Arousal

<table>
<thead>
<tr>
<th>Arousal classes</th>
<th>Emotion / State</th>
<th>Data samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Calm</td>
<td>Sadness, Boredom</td>
<td>15</td>
</tr>
<tr>
<td>Medium Arousal</td>
<td>Happiness</td>
<td>16</td>
</tr>
<tr>
<td>Excited / Activated</td>
<td>Surprise, Excitement,</td>
<td>57</td>
</tr>
<tr>
<td></td>
<td>Frustration, Anger, Concentration</td>
<td></td>
</tr>
</tbody>
</table>

TABLE VI: Emotions and States Mapped in Classes on Valence

<table>
<thead>
<tr>
<th>Valence classes</th>
<th>Emotion / State</th>
<th>Data samples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unpleasant</td>
<td>Sadness, Boredom, Frustration, Anger</td>
<td>36</td>
</tr>
<tr>
<td>Neutral Valence</td>
<td>Surprise, Concentration</td>
<td>30</td>
</tr>
<tr>
<td>Pleasant</td>
<td>Happiness, Excitement</td>
<td>22</td>
</tr>
</tbody>
</table>

CONCLUSIONS

Affect detection is indeed a challenging task. Databases such as FEEDB provide valuable information to researchers, and constitute a useful experimental basis for data analysis. In this paper, we have presented an extension to the database. Data has been extracted from recordings in a proprietary format (XED), by using the Kinect for Windows SDK. In particular, we have extracted data related to 100 facial features, the
position of the user’s head and the head pose angles, along with 6 AU and 11 SU according to a Candi 3 model [14]. This additional data has been provided for a total of 88 subject-emotion combinations, each in separate text files. This is a general approach and can be applied to any collection of XED files. This makes it possible to use the data without a Microsoft Kinect Sensor attached to the machine.

ACKNOWLEDGEMENTS
This work has been partly supported by the Spanish Ministry of Economy and Competitiveness through projects TIN2011-29221-C03-02 and TIN2014-59641-C2-1-P; and by the Polish-Norwegian Research Programme operated by the National Centre for Research and Development under the Norwegian Financial Mechanism 2009-2014 in the frame of Project Contract No Pol-Nor/210629/51/2013.

REFERENCES
AUTHOR LISTING
<table>
<thead>
<tr>
<th>Author</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abdallah R.B.</td>
<td>107</td>
</tr>
<tr>
<td>Abdouli M.</td>
<td>42/318</td>
</tr>
<tr>
<td>Abelha A.</td>
<td>64</td>
</tr>
<tr>
<td>Aboutaleb A.</td>
<td>47</td>
</tr>
<tr>
<td>Adams C.</td>
<td>47</td>
</tr>
<tr>
<td>Ahmed A.B.</td>
<td>172</td>
</tr>
<tr>
<td>Al-Azawi R.</td>
<td>346</td>
</tr>
<tr>
<td>Albert V.</td>
<td>31</td>
</tr>
<tr>
<td>Aleiyouka M.</td>
<td>42</td>
</tr>
<tr>
<td>Aliyu H.O.</td>
<td>141/149/193</td>
</tr>
<tr>
<td>Alpan G.</td>
<td>117</td>
</tr>
<tr>
<td>Alpers S.</td>
<td>357</td>
</tr>
<tr>
<td>Alves J.C.P.</td>
<td>57</td>
</tr>
<tr>
<td>Amanton L.</td>
<td>42</td>
</tr>
<tr>
<td>Arevalillo-Herráez M.</td>
<td>455</td>
</tr>
<tr>
<td>Arsene R.</td>
<td>273</td>
</tr>
<tr>
<td>Atorf L.</td>
<td>385</td>
</tr>
<tr>
<td>Ayesh A.</td>
<td>346/455</td>
</tr>
<tr>
<td>Baffa M.F.O.</td>
<td>341</td>
</tr>
<tr>
<td>Balsamo S.</td>
<td>81</td>
</tr>
<tr>
<td>Becker C.</td>
<td>357</td>
</tr>
<tr>
<td>Berndt O.</td>
<td>315</td>
</tr>
<tr>
<td>Berred A.</td>
<td>42</td>
</tr>
<tr>
<td>Bertelle C.</td>
<td>323</td>
</tr>
<tr>
<td>Birzniece I.</td>
<td>36</td>
</tr>
<tr>
<td>Bogdan S.</td>
<td>91</td>
</tr>
<tr>
<td>Bontos M.D.</td>
<td>99</td>
</tr>
<tr>
<td>Bouaziz R.</td>
<td>318</td>
</tr>
<tr>
<td>Brandau C.</td>
<td>122</td>
</tr>
<tr>
<td>Brandmeier T.</td>
<td>297</td>
</tr>
<tr>
<td>Brito L.M.P.L.</td>
<td>410</td>
</tr>
<tr>
<td>Brodziak R.</td>
<td>225</td>
</tr>
<tr>
<td>Bylka J.</td>
<td>225</td>
</tr>
<tr>
<td>Călinoiu C.</td>
<td>99</td>
</tr>
<tr>
<td>Cappart Q.</td>
<td>402</td>
</tr>
<tr>
<td>Cardoso R.P.P.</td>
<td>399</td>
</tr>
<tr>
<td>Caris A.</td>
<td>428</td>
</tr>
<tr>
<td>Carvalho F.M.M.</td>
<td>22</td>
</tr>
<tr>
<td>Casimiro A.D.</td>
<td>399</td>
</tr>
<tr>
<td>Charrier R.</td>
<td>323</td>
</tr>
<tr>
<td>Chetto M.</td>
<td>164/254</td>
</tr>
<tr>
<td>Chniter H.</td>
<td>164</td>
</tr>
<tr>
<td>Choi S.Y.</td>
<td>289</td>
</tr>
<tr>
<td>Cichon T.</td>
<td>385</td>
</tr>
<tr>
<td>Clausen U.</td>
<td>423</td>
</tr>
<tr>
<td>Clement R.</td>
<td>433</td>
</tr>
<tr>
<td>Clua E.W.G.</td>
<td>341</td>
</tr>
<tr>
<td>Conley W.</td>
<td>5/10</td>
</tr>
<tr>
<td>Correia L.</td>
<td>91</td>
</tr>
<tr>
<td>Costa M.</td>
<td>205</td>
</tr>
<tr>
<td>Croitorescu V.</td>
<td>297</td>
</tr>
<tr>
<td>da Costa Rodrigues B.</td>
<td>57</td>
</tr>
<tr>
<td>da Silva Bento D.</td>
<td>57</td>
</tr>
<tr>
<td>Danielsson F.</td>
<td>363/377</td>
</tr>
<tr>
<td>de Mello B.A.</td>
<td>22</td>
</tr>
<tr>
<td>de Paulo L.M.</td>
<td>57</td>
</tr>
<tr>
<td>de Sousa B.L.</td>
<td>57</td>
</tr>
<tr>
<td>Deuse J.</td>
<td>371</td>
</tr>
<tr>
<td>Díaz G.</td>
<td>73</td>
</tr>
<tr>
<td>Diekmann D.</td>
<td>423</td>
</tr>
<tr>
<td>Djitog I.</td>
<td>193</td>
</tr>
<tr>
<td>Dormuth I.</td>
<td>423</td>
</tr>
<tr>
<td>Duffy A.</td>
<td>47/433</td>
</tr>
<tr>
<td>Dupleac D.</td>
<td>273</td>
</tr>
<tr>
<td>Duvallet C.</td>
<td>318</td>
</tr>
<tr>
<td>Embley T.</td>
<td>47</td>
</tr>
<tr>
<td>Eryilmaz E.</td>
<td>357</td>
</tr>
<tr>
<td>Faustino P.R.C.</td>
<td>341</td>
</tr>
<tr>
<td>Filho C.C.</td>
<td>205</td>
</tr>
<tr>
<td>Furian N.</td>
<td>185/197</td>
</tr>
<tr>
<td>Garcia-Fernández I.</td>
<td>211</td>
</tr>
<tr>
<td>Ghribi I.</td>
<td>107</td>
</tr>
<tr>
<td>Glenn J.</td>
<td>47</td>
</tr>
<tr>
<td>Glorieux E.</td>
<td>377</td>
</tr>
<tr>
<td>Gomes J.</td>
<td>64</td>
</tr>
<tr>
<td>Greenwood A.G.</td>
<td>117</td>
</tr>
<tr>
<td>Gremzow C.</td>
<td>122</td>
</tr>
<tr>
<td>Guerfala E.</td>
<td>302</td>
</tr>
<tr>
<td>Guerin F.</td>
<td>237</td>
</tr>
<tr>
<td>Harrouch H.</td>
<td>318</td>
</tr>
<tr>
<td>Heinze F.</td>
<td>371</td>
</tr>
<tr>
<td>Herbez C.</td>
<td>133</td>
</tr>
<tr>
<td>Hopwell A.</td>
<td>433</td>
</tr>
<tr>
<td>Housseyni W.</td>
<td>254</td>
</tr>
<tr>
<td>Huber L.</td>
<td>315</td>
</tr>
<tr>
<td>Ina C.</td>
<td>99</td>
</tr>
<tr>
<td>Ivaschenko A.</td>
<td>333</td>
</tr>
<tr>
<td>Janssens G.K.</td>
<td>428</td>
</tr>
<tr>
<td>Jaray F.</td>
<td>164</td>
</tr>
<tr>
<td>Jurenoks A.</td>
<td>245</td>
</tr>
<tr>
<td>Author Name</td>
<td>Page Number</td>
</tr>
<tr>
<td>-----------------------------</td>
<td>-------------</td>
</tr>
<tr>
<td>Kaddes M.</td>
<td>42/318</td>
</tr>
<tr>
<td>Kalibatiene D.</td>
<td>36</td>
</tr>
<tr>
<td>Kang H.I.</td>
<td>284/289</td>
</tr>
<tr>
<td>Kang P.S.</td>
<td>47/433</td>
</tr>
<tr>
<td>Karoui O.</td>
<td>302</td>
</tr>
<tr>
<td>Khalgui M.</td>
<td>107/164/172</td>
</tr>
<tr>
<td></td>
<td>254/302</td>
</tr>
<tr>
<td>Kim H.S.</td>
<td>284/289</td>
</tr>
<tr>
<td>Klöckner M.</td>
<td>371</td>
</tr>
<tr>
<td>Kolsanov A.</td>
<td>333</td>
</tr>
<tr>
<td>Kouba A.</td>
<td>302</td>
</tr>
<tr>
<td>Kuhlenkoetter B.</td>
<td>371</td>
</tr>
<tr>
<td>Kuhnert K.-D.</td>
<td>392</td>
</tr>
<tr>
<td>Kurdi O.</td>
<td>441</td>
</tr>
<tr>
<td>Kuzmin A.</td>
<td>333</td>
</tr>
<tr>
<td>Ladier A.-L.</td>
<td>117</td>
</tr>
<tr>
<td>Lefebre D.</td>
<td>237</td>
</tr>
<tr>
<td>Legato P.</td>
<td>418</td>
</tr>
<tr>
<td>Legay A.</td>
<td>402</td>
</tr>
<tr>
<td>Lennartson B.</td>
<td>377</td>
</tr>
<tr>
<td>Limbrée C.</td>
<td>402</td>
</tr>
<tr>
<td>Lozano M.</td>
<td>211</td>
</tr>
<tr>
<td>Machado A.F.V.</td>
<td>57/341</td>
</tr>
<tr>
<td>Machado J.</td>
<td>64</td>
</tr>
<tr>
<td>Macià H.</td>
<td>73</td>
</tr>
<tr>
<td>Maes T.</td>
<td>428</td>
</tr>
<tr>
<td>Maiga O.</td>
<td>141/149</td>
</tr>
<tr>
<td>Marco-Giménez L.</td>
<td>455</td>
</tr>
<tr>
<td>Mariano P.</td>
<td>91</td>
</tr>
<tr>
<td>Marin A.</td>
<td>81</td>
</tr>
<tr>
<td>Martínez-Durá R.J.</td>
<td>211</td>
</tr>
<tr>
<td>Mateescu C.T.</td>
<td>273</td>
</tr>
<tr>
<td>Mazza R.M.</td>
<td>418</td>
</tr>
<tr>
<td>Menezes M.</td>
<td>205</td>
</tr>
<tr>
<td>Miklić D.</td>
<td>91</td>
</tr>
<tr>
<td>Mills R.</td>
<td>91</td>
</tr>
<tr>
<td>Moreira G.B.S.M.</td>
<td>341</td>
</tr>
<tr>
<td>Mosbah O.</td>
<td>172/254</td>
</tr>
<tr>
<td>Muakad C.F.J.</td>
<td>410</td>
</tr>
<tr>
<td>Müller K.</td>
<td>392</td>
</tr>
<tr>
<td>Nawayseh N.</td>
<td>278</td>
</tr>
<tr>
<td>Nazaryan A.</td>
<td>333</td>
</tr>
<tr>
<td>Ndiaye M.F.</td>
<td>237</td>
</tr>
<tr>
<td>Ndiaye P.A.</td>
<td>237</td>
</tr>
<tr>
<td>Neubacher D.</td>
<td>185/197</td>
</tr>
<tr>
<td>Neves Júnior A.B.</td>
<td>57</td>
</tr>
<tr>
<td>Ngwompo R.F.</td>
<td>16</td>
</tr>
<tr>
<td>Nicolici S.</td>
<td>273</td>
</tr>
<tr>
<td>Nocickis L.</td>
<td>245</td>
</tr>
<tr>
<td>Nowak M.</td>
<td>265</td>
</tr>
<tr>
<td>Oliveira A.L.C.</td>
<td>410</td>
</tr>
<tr>
<td>O'Sullivan M.</td>
<td>197</td>
</tr>
<tr>
<td>Petriczek G.</td>
<td>229</td>
</tr>
<tr>
<td>Platzner M.</td>
<td>107</td>
</tr>
<tr>
<td>Ponnusamy S.S.</td>
<td>31</td>
</tr>
<tr>
<td>Portela F.</td>
<td>64</td>
</tr>
<tr>
<td>Potthoff N.</td>
<td>122</td>
</tr>
<tr>
<td>Prisecaru I.</td>
<td>273</td>
</tr>
<tr>
<td>Quesnel G.</td>
<td>133</td>
</tr>
<tr>
<td>Rabai H.</td>
<td>323</td>
</tr>
<tr>
<td>Radev D.</td>
<td>159</td>
</tr>
<tr>
<td>Radeva S.</td>
<td>149</td>
</tr>
<tr>
<td>Ramaekers K.</td>
<td>428</td>
</tr>
<tr>
<td>Ramalho M.M.</td>
<td>341</td>
</tr>
<tr>
<td>Ramat E.</td>
<td>133</td>
</tr>
<tr>
<td>Reddy D.</td>
<td>363</td>
</tr>
<tr>
<td>Riezboz S.</td>
<td>197</td>
</tr>
<tr>
<td>Rima A.</td>
<td>36</td>
</tr>
<tr>
<td>Rodríguez-Cerro A.</td>
<td>211</td>
</tr>
<tr>
<td>Romano D.M.</td>
<td>441</td>
</tr>
<tr>
<td>Rosa P.</td>
<td>448</td>
</tr>
<tr>
<td>Rossetti R.J.F.</td>
<td>399/448</td>
</tr>
<tr>
<td>Rossmann J.</td>
<td>371/385</td>
</tr>
<tr>
<td>Rudzajs P.</td>
<td>36</td>
</tr>
<tr>
<td>Salas J.M.</td>
<td>16</td>
</tr>
<tr>
<td>Sampaio P.N.M.</td>
<td>410</td>
</tr>
<tr>
<td>Santos M.F.</td>
<td>64</td>
</tr>
<tr>
<td>Schaus P.</td>
<td>402</td>
</tr>
<tr>
<td>Silva J.P.S.</td>
<td>410</td>
</tr>
<tr>
<td>Smielik I.</td>
<td>392</td>
</tr>
<tr>
<td>Spinola S.S.</td>
<td>410</td>
</tr>
<tr>
<td>Sroczan E.M.</td>
<td>251</td>
</tr>
<tr>
<td>Stannett M.</td>
<td>441</td>
</tr>
<tr>
<td>Stojic I.</td>
<td>81</td>
</tr>
<tr>
<td>Studzinski J.</td>
<td>219/229</td>
</tr>
<tr>
<td>Svensson B.</td>
<td>363/377</td>
</tr>
<tr>
<td>Szwoch M.</td>
<td>455</td>
</tr>
<tr>
<td>Thebault P.</td>
<td>31</td>
</tr>
<tr>
<td>Tomasiuk D.</td>
<td>219</td>
</tr>
</tbody>
</table>
AUTHOR LISTING

Traoré M.K. .................. 141/149/193
Tuca A. ...................... 297
Tutsch D. .................... 122

Urbaniak A. ................. 225/265

Valero V. .................... 73
Vasilicas O. ................ 36
Vasiliiu D. .................. 99
Vasiliiu N. .................. 99
Vössner S. .................. 185/197

Walker C. ................... 197

Zakrzewski P. ............... 225
Ziółkowski A. ............... 219